Self-Understanding and Self-Extension: A Systems and Representational Approach

Jeremy L. Wyatt, Alper Aydemir, Michael Brenner, Marc Hanheide, Nick Hawes, Patric Jensfelt, Matej Kristan, Geert-Jan M. Kruijff, Pierre Lison, Andrzej Pronobis, Kristoffer Sjöö, Alen Vrečko, Hendrik Zender, Michael Zillich, and Danijel Skočaj

***Abstract*—There are many different approaches to building a system that can engage in autonomous mental development. In this paper, we present an approach based on what we term *self-understanding*, by which we mean the explicit representation of and reasoning about what a system does and does not know, and how that knowledge changes under action. We present an architecture and a set of representations used in two robot systems that exhibit a limited degree of autonomous mental development, which we term *self-extension*. The contributions include: representations of gaps and uncertainty for specific kinds of knowledge, and a goal management and planning system for setting and achieving learning goals.**

***Index Terms*—Architectures, representations, robot learning, robotics.**

I. Introduction

WHAT IS needed for an agent to learn in a truly autonomous fashion? Autonomous learning requires that the agent pick its own learning goals. One way to achieve this is to give that agent representations of what it knows and does not know, and to make it reason with these representations to set its own *epistemic goals*. An epistemic goal is a goal to be in a certain knowledge state. This paper describes this approach to autonomous mental development. We present an architecture, together with a set of representations that explicitly capture what the robot and other agents do and do not know at any time, i.e., representations of their epistemic state. We also describe representations of how this epistemic state will change under action. Such representations, together with algorithms for reasoning about them confer a degree of *self-understanding*, and allow the agent to plan how to extend its abilities, or knowledge of the environment, i.e., *self-extension*. We also describe a goal management system that allows the robot to choose quickly between different epistemic goals. This mechanism is necessary to allow our approach to scale, since if a robot generates many possible learning goals the time taken to plan for them all will be too great.

We first define self-understanding and self-extension as we see them. To do this it is necessary to characterize the different types of incompleteness in knowledge that will be represented. We use *incompleteness* as an umbrella term to cover many different types of *knowledge gaps* and *uncertainty about knowledge*. We can construct a typology of incompleteness in knowledge-based on three dimensions of variability. These are *the nature of the incompleteness*, the *type of knowledge that is incomplete*, and whether the incompleteness is represented in a *quantitative or qualitative* manner.

With regard to the nature of the incompleteness, in the simplest case, we may have a variable or variables that are part of a model of the world and which have a defined set of possible values or hypotheses from which the true value is known to be drawn. We refer to this as *simple uncertainty*. We can also have *uncertainty about the number of variables* needed in a model, i.e., about the model complexity. Finally, we can also have cases where the agent knows that a variable is of an unexperienced class, i.e., there is *novelty*. This can include cases where the variables are continuous, but where the observation models for a class are quite confident and do not generalize well to some new observation. The type of knowledge that is incomplete may vary enormously. Five simple types that cover a variety of cases include contingent knowledge about the current world *state*, *structural* knowledge about the universal relationships between variables, *procedural* knowledge about how to act in certain situations to achieve certain goals, knowledge consisting of *predictions* of action outcomes or events, and knowledge about their *causes*. Finally, there is a question about whether the representation is qualitative or quantitative. In qualitative representations, we simply have a set of possible values for the variable, or a statement that the variable value is unknown, or knowledge that there may be many variables that are unmodeled. In quantitative representations, we will have some kind of scalar values attached to hypotheses (such as whether there is novelty or not), and in our case these will typically be probabilities. Note that by a *quantitative gap* or *quantitative uncertainty*, we do not mean

Manuscript received March 01, 2010; revised September 13, 2010; accepted October 12, 2010. Date of publication November 01, 2010; date of current version December 10, 2010. This work was supported in part by the EC FP7 IST Project CogX-215181.

J. L. Wyatt, M. Hanheide, and N. Hawes are with the School of Computer Science, University of Birmingham, Birmingham, B15 2TT, U.K. (e-mail: jlw@cs.bham.ac.uk; m.hanheide@cs.bham.ac.uk; nah@cs.bham.ac.uk).

A. Aydemir, P. Jensfelt, A. Pronobis, and K. Sjöö are with the Centre for Autonomous Systems, KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: aydemir@csc.kth.se; patric@csc.kth.se; pronobis@csc.kth.se; krsj@csc.kth.se).

M. Brenner is with Albert-Ludwigs-Universität, Freiburg, 79098, Germany (e-mail: brenner@informatik.uni-freiburg.de).

M. Kristan, A. Vrečko, and D. Skočaj are with University of Ljubljana, Ljubljana, Slovenia (e-mail: matej.kristan@fri.uni-lj.si; alen.vrecko@fri.uni-lj.si; danijel.skocaj@fri.uni-lj.si).

G.-J. M. Kruijff, P. Lison, and H. Zender are with DFKI, Saarbrücken, 66123, Germany (e-mail: gj@dfki.de; plison@dfki.de; zender@dfki.de).

M. Zillich is with Vienna University of Technology, 1040 Vienna, Austria (e-mail: zillich@acin.tuwien.ac.at).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAMD.2010.2090149

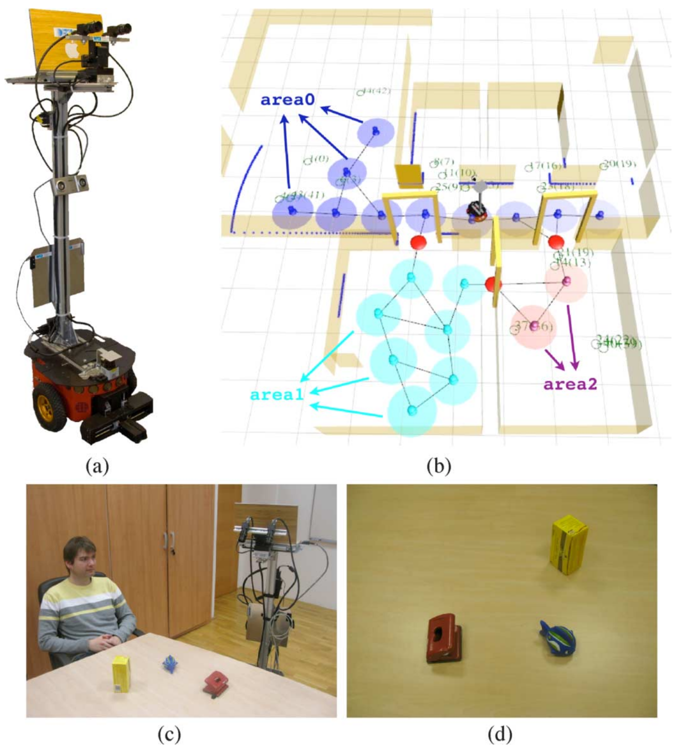


Fig. 1. Dora the Explorer (top row) and George (bottom row). (a) The Dora platform: a P3 mobile robot base with a custom-built super-structure and different sensors. (b) Visualization of Dora's map of a partially explored environment. Colored disks denote *place* nodes (color indicates segmentation into different rooms, `area0`, `1`, `2`). Small green circles represent opportunities for spatial exploration (*placeholders*). Red nodes indicate places where doorways are located. (c) George: Scenario setup. (d) George: Observed scene.

that the underlying space for the variable is continuous or discrete, but instead that the way the incompleteness is represented involves an expression of a degree of preference for one hypothesis or statement versus another.

Given this brief characterization of different types of incompleteness, we can define self-understanding and self-extension compactly as follows. A system with *self-understanding* is any system with explicit representations that captures some kind(s) of incompleteness in some kind(s) of knowledge that it possesses. A *self-extending* system is a system that is able to fill in this incomplete knowledge. Note that a self-extending system may not necessarily be one that self-understands.

In this paper, we describe systems that have a limited degree of self-understanding and perform self-extension by exploiting this self-understanding. Specifically in this paper, we deal with filling qualitative gaps, qualitative uncertainty in state, quantitative uncertainty about structural knowledge, and novel states. We provide empirical proof that our approach works and illustrate different aspects of it through two robot systems we have implemented. We call these Dora, and George (see Fig. 1 (a), (b) and (c), (d), respectively). We provide links to videos of these systems running in the real world,[1] and analyze their behavior on larger amounts of data using simulations. We now describe Dora and George in more detail to give concrete examples that will run through the paper.

[1]Available at http://cogx.eu/

Dora is able to explore an office environment, choosing between filling different types of gaps in her map of the environment. Dora illustrates the architecture, the representations of gaps in spatial knowledge, the use of epistemic states in goal setting and action planning, and the use of our motivation system. In Dora's case, the current incomplete knowledge she can model and fill can be seen by reference to Fig. 1(b). Here we can see a visualization of a map that Dora has built after a partial tour of an office environment. The map consists of a graph where the nodes (which we call *places*) are partitioned into areas by landmarks, such as doors. Dora has representations of two kinds of incompleteness. She represents unexplored regions of space, by maintaining a set of hypothesized places, which we call *placeholders*. These are depicted in Fig. 1(b) as small unfilled circles with numeric labels. This is uncertainty about how many variables are needed to model the space. Second, Dora has the ability to categorize areas into categories such as office, kitchen, coffee room, and corridor. In Fig. 1(b), it can be seen that none of the areas currently have known categories. This is simple state uncertainty, as Dora knows a certain number of types of area, and cannot represent or reason about novel area types. During the autonomous part of the mapping process, Dora will choose the order in which to reduce these different kinds of incompleteness. To map unexplored areas by adding nodes to the topological map, she will conduct laser-based mapping while visiting the hypothesized placeholders. To categorize a room, she will search for objects that indicate its category, e.g., kitchens typically contain objects such as milk cartons and cups, and offices contain objects such as bookshelves and journals.

George is a system that converses with a human to reduce incompleteness it has about the properties of objects on a table top. George illustrates the way we represent uncertainty and incompleteness in models of the structural relationships between visual information and linguistic descriptions of objects. What visual properties, for example, make an object round or square versus red or yellow? George has representations of the uncertainty it has as to which subspaces of a set of visual features are associated with particular adjectives. This is a type of structural uncertainty. George can learn from a tutor, but crucially he can also decide which questions to ask in order to fill a particular gap he has identified. A typical scene during George's learning is depicted in Fig. 1(c). A typical dialog might be

A: What color is the elongated object?
B: The elongated object is yellow.
C: OK.
D: Is the square object blue?
E: No it is not blue. It is red.
F: OK.

During this dialog, the robot and the human reason about each other's beliefs, what they know and do not know, and how to establish common understanding. This is a type of state uncertainty, since the robot can only model the human as having one of a known set of beliefs. In the dialog, each agent makes references to objects that they understand will be distinguishing to the other agent, such as referring to the elongated object. More importantly, George asks questions which are prompted by detection of gaps such as state novelty. He asks: *"Which color is ...?"* when he realizes that the color is one he has not experi-

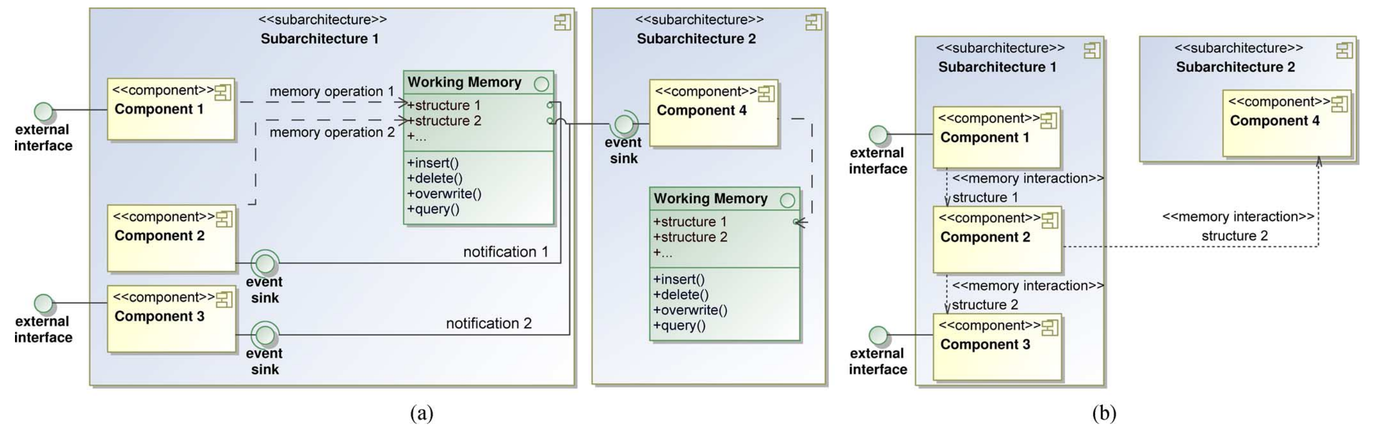


Fig. 2. Building systems with CAS. The form on the right is used as a short hand in the later system diagram for Dora. (a) In CAS, there are components, which run in parallel, asynchronously updating shared structures on a common working memory. They can also take input from sensors or give output to actuators. Subarchitectures are coupled to make an overall system. (b) A simplified illustration of the interaction patterns mediated through the working memories focusing on information flow from sources to sinks.

enced before. When he is simply uncertain about which of a number of couloir classes is present, he asks instead whether the object has the most likely color class: *Is the object blue?*. Both George and Dora also have mechanisms for doing non-monotonic inference or learning. George can unlearn erroneous representations of color and shape, and in Dora's case she can withdraw support for inferences about room category, or the partition of her map.

The rest of the paper is structured as follows. In Section II, we describe the architectural model. Section III describes the model that connects information from multiple modalities, and how we have engineered those representations to explicitly capture uncertainty and incompleteness in the amodal model. Section IV covers representations of space, cross-modal relations, epistemic goals, and epistemic action effects, all as used in Dora and/or George. In Section V, we describe our approach to goal management, and in Sections VI and VII, we describe the Dora and George systems and present an experimental analysis of their behavior. The paper closes with a summary of related approaches and a discussion.

II. An Architecture for Multimodal Processing

In this section, we describe the basic architecture we employ, called CoSy architecture achema (CAS) [1]. It is a schema because it actually defines a space of architectures, so to be clear, we refer to the schema when talking about that space, and to an architecture when we mean a specific architecture used in a robot system. The schema is essentially a distributed working memory model, where representations are linked within and across the working memories, and are updated asynchronously and in parallel. The key idea is that it replaces a single world model (still prevalent in robotics) with multiple, linked world models, enabling it to work in the face of inconsistent evidence, uncertainty, and change. The decomposition into working memories groups processes that commonly share information, and is typically by sensory modality. So that in Dora and George, we build separate subsystems (called *subarchitectures*) for vision, communication, and spatial understanding. As we will see in Sections III and IV, each subarchitecture contains representations which explicitly capture uncertainty and incompleteness. The system overall can reason about this uncertainty or incompleteness and plan how to act so as to fill that knowledge gap, perhaps by employing information in another modality. We now describe the key aspects of CAS relevant to this paper.

A. Subarchitecture Design

1) Components: Our schema starts on the level of a collection of processing components [see Fig. 2(a)], which can process concurrently. We do not specify any constraints on the contents of components. They could have behave like a node in a connectionist network, an activity in a behavior-based system, or an entire function in a functionally composed system. Components can take input either directly from sensors or from working memory. They can also directly control actuators in the manner of closed loop controllers, or initiate fixed action patterns. Components can have processing triggered by the appearance of certain information on the shared working memory, and can modify structures on that memory. Components may also have their own private (i.e., component-internal) memory.

2) Shared Working Memories: Rather than exchange information directly, processing components are connected to a *shared working memory* [see Fig. 2(a)]. The contents of the working memory are solely composed of the outputs of processing components. Each working memory is connected to all other working memories in the system. This allows components to exchange information across subarchitectures. In our implementation of CAS, the communication method between the working memory and the components determines the efficiency of the model. But for now, let us consider simply that the schema itself allows reading and writing to working memories, and transfer of information between them.

This use of shared working memories is particularly well suited to the *collaborative refinement of shared structures*. In this approach to information processing, a number of components use the data available to them to incrementally update a shared data structure on working memory. In this manner, the results of processing done by one component can affect that done

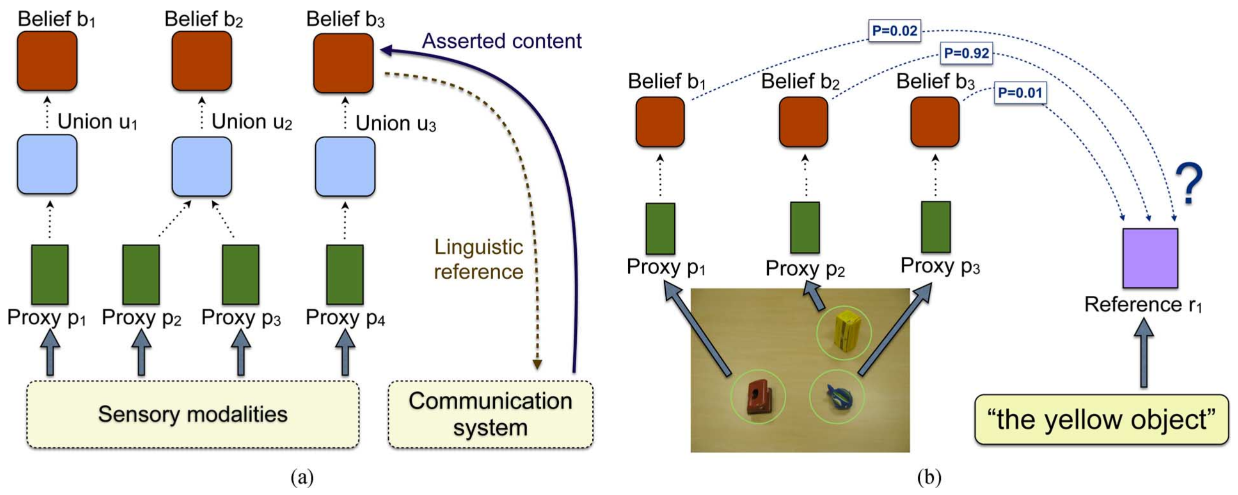


Fig. 3. Multimodal information binding: belief construction (left) and application in a reference resolution task (right). (a) Construction of multimodal beliefs. (b) Reference resolution for the expression "the yellow object."

by others. As all components are active in parallel, the collective total processing effort may be reduced by sharing information in this way. This feature turns out to be a very powerful aspect of the schema.

B. System Design Practices With CAS

While a system could be composed of a single subarchitecture, there are typically several. The schema makes no assumptions about whether system decomposition should be functional or behavioral. What is important is that the decomposition groups components that commonly exchange information via shared structures. One of the main benefits of having distributed working memories is that each working memory can act as a filter for its components, only letting them become aware of events elsewhere in the system when necessary.

The systems described here (Dora and George) have subarchitectures for vision, communication, and spatial understanding, each of which builds its own partial model of the world. To have coherent global action, however, we must link these separate models. For example, the visual system might have a symbol corresponding to a blue cup sitting on a table, and the communication system might have a symbol corresponding to a mention of a blue thing by a human. To make sense of the human's words the robot has to decide whether these two symbols are connected, or whether the human is referring to some other object (perhaps another blue object in the scene). An important question is what mechanisms can be used to link these symbols efficiently? We call this symbol linking problem the *binding problem* and we describe it in the next section. In particular, we describe how the binding problem is solved in a way that explicitly represents the uncertainty in which symbols should be bound to which, and the gaps in the robot's overall model of the world. Binding essentially involves creating new symbols that refer back to each of the modality specific symbols. Because of this we refer to the complex representations that are created by binding as multimodal representations. At the highest level of abstraction, however, binding produces an essentially amodal model of the robot's world.

C. Architectures: Related Work

The architecture described here builds on that in [2]. Its mix of multiple subarchitectures, multimodal beliefs formed via binding, and continual planning for control is novel. Support for modality specific representations and true parallel processing sets it apart from architectures such as Soar [3] and ACT-R [4], which use unified representation schemes and mostly process information in a single thread. We argue that these additional properties of CAS are essential for progress in cognitive robotics. The DIARC architecture [5] is closest to our approach, using a planner to drive robot behavior, with multiple concurrent subsystems for different functionalities, but it does not do explicit belief modeling or represent gaps in the system's knowledge. Another archictecture schema similar to CAS is the memory-based architecture used in BIRON [6].

III. Modeling Multimodal Beliefs

So far we have described an architecture capable of distributed processing of modality specific representations. High-level cognitive abilities need to operate on abstract representations that summarize and integrate information from these. This requirement raises two issues: 1) how abstract representations can be reliably generated from many unreliable low-level representations; and 2) how distributed information can be efficiently fused into multimodal structures.

Here, we present a Bayesian approach to solve both problems. The approach is implemented in a specific subarchitecture called the *binder* [7]. The binder is directly connected to all other subarchitectures. It is a central hub for the information gathered about entities currently perceived in the environment. The information on the binder is inherently *probabilistic*, i.e., each piece of information is given a probability value, reflecting the confidence level of the subarchitecture that generated it.

Based on the data structures in this hub, the binding algorithm seeks to "merge" or unify the perceptual inputs arising from the various subsystems, by checking whether their respective features correlate with each other. The probability of these correlations are encoded in a Bayesian network. This network might for instance express a high compatibility between the haptic feature "shape: cylindrical" and the visual feature "object: mug" (since most mugs are cylindrical), but a low compatibility between the features "shape: cylindrical" and "object: ball."

When two modality specific structures are bound, the resulting multimodal information structure is called a *belief* in our terminology. The task of the binder is to decide which structures from different modalities refer to the same real-world entity, and should therefore be merged into a belief. The outcome of this process is a joint probability distribution over possible beliefs. These beliefs are a compact summary of the information distributed across other subarchitectures.

A. Representations

The three central data structures manipulated by the binder are **proxies**, **unions**, and **beliefs** [also see Fig. 3(a)].

1) Proxies: A midlevel, unimodal representation of a given entity in the environment. Proxies are inserted onto the binder by the various subarchitectures. A proxy is defined as a multivariate probabilistic distribution over a set of features. The distributions in the proxy can be discrete (as for categorical knowledge) or continuous.

2) Unions: A midlevel, multimodal representation of an entity, constructed by merging one or more proxies. Just like proxies, unions are also represented as a multivariate probabilistic distribution over possible features. Unions are a transitional layer between proxies and beliefs.

3) Beliefs: A high-level, amodal representation of an entity in the environment. Beliefs are generally build on top of unions, but they are expressed in an amodal format and encode additional information related to the specific situation and perspective in which the belief was formed, such as its *spatio–temporal frame*, its *epistemic status* and its *saliency value*:

- The spatio–temporal frame is defined according to a spatial model (set of possible "places" in the environment), a temporal model (points on a continuous temporal interval), and possibly a perspective on these two models from the viewpoint of a particular agent.
- The epistemic status of a belief (or subpart of a belief) can be either private, attributed, or shared. Private beliefs are beliefs which are internal to the agent, while attributed beliefs are beliefs an agent ascribes to another agent (e.g., A believes that B believes X). Shared beliefs are beliefs which are part of the common ground for all agents.
- Finally, the salience is a multivariate density function $\Re^n \rightarrow [0,1]$, where each variable defines a particular, real-valued saliency measure. It provides an estimate of the "importance" or quality of standing out of a particular entity relative to neighboring ones [8]. The salience is used to drive the attentional behavior of the agent by specifying which entities are currently in focus.

Beliefs are indexed via an unique identifier, which allows us to keep track of the whole development history of a particular belief. Beliefs can also be connected with each other using relational structures of arbitrary complexity. To account for this rich representation, beliefs are formalized according to a *belief model*, which is a mathematical structure defining a space of possible belief instances.

B. Binding Algorithm

To be able to create beliefs out of proxies, the binder must decide for each pair of proxies arising from distinct subsystems, whether they should be bound into a single union, or placed in two separate unions. The decision algorithm for this task is based on a well-known technique from probabilistic data fusion, called the *independent likelihood pool* (ILP) [9]. Using the ILP, we are able to compute the likelihood of every possible binding of proxies, and use this estimate as a basis for constructing the beliefs. The multivariate probability distribution contained in the belief is a linear function of the feature distributions included in the proxies and the correlations between these.

A Bayesian network encodes all possible feature correlations as conditional dependencies. The encoded features may be discrete or continuous. The (normalized) product of these correlations over the complete feature set provides an estimate of the "internal consistency" of the constructed belief—a belief with incompatible features will have a near-zero probability, while a belief with highly correlated features will have a high probability.

C. Referencing and Updating Beliefs

The beliefs are high-level symbolic representations available to the whole system. They provide an unified model of the environment which can be used when interacting with the human user. An important aspect of this is *reference resolution*: how to connect linguistic expressions such as "this box" or "the ball on the floor" to the corresponding beliefs about entities in the environment.

Reference resolution is performed using the same core mechanisms as for binding—a Bayesian network specifies the correlations between the linguistic constraints of the referring expressions and the belief features (in particular, the entity saliency and associated categorical knowledge). The resolution process yields a probability distribution over alternative referents [see, for example, Fig. 3(b)], which is then retrieved by the communication subsystem for further interpretation.

In addition to simple reference, the interaction with a human user can also provide *new* content to the beliefs, as in cross-modal learning scenarios. Via (linguistic) communication, the human user can thus, directly extend or modify the robot's current beliefs. If this new information conflicts with existing perceptual knowledge, the agent can trigger a clarification request to resolve the conflict.

An utterance such as "This is yellow" illustrates these two complementary mechanisms. First, the linguistic expression "this" must be resolved to a particular entity in the environment. Since "this" is a (proximal) deictic, the resolution is performed on basis of the saliency measures. In the absence of any other constraint, the most salient entity is simply selected and retrieved. Second, the utterance not only refers to an existing entity in the environment, but it also provides new information about it—namely that it is yellow. This asserted content must therefore inserted into the robot's beliefs. This is realized by selecting the belief pertaining to the referred-to entity and incorporating the new, attributed information into its content representation.

Finally it is important to note that the Bayesian network encoding the feature correlations can be either manually specified, or learned using various machine learning techniques (see Section VII). This latter approach corresponds to self-extension of the robot's categorical knowledge.

D. *Multimodal Beliefs: Related Work*

There is a variety of approaches to multimodal information binding (see [9] and [10]). In [11], Roy introduces *semiotic schemas*, which are abstract representations encoding connections across sensory-motor modalities, but he does not describe how the structure and parameters associated with these are specified or learned. It is also worth noting that, contrary to purely symbolic approaches such as [7], all data structures manipulated by the binder are probabilistic representations, enabling the system to deal explicitly with the uncertainty unavoidable for most perceptual processes. Furthermore, the encoding of cross-modal correlation rules (used for, e.g., reference resolution) as a Bayesian network makes it possible to *learn* the model parameters. Another distinctive feature of our approach is the use of a single, shared information repository instead of point-to-point connections between modalities. Such shared repository allows us to centralize the robot's current knowledge at a single location in the global system architecture, and directly perform cross-modal binding operations over the data structures defined in this repository.

IV. Representations to Support Self-Extension

This section explains how different domains of knowledge are represented in our system. We discuss three: 1) representations of space, used in the Dora scenario; 2) cross-modal representations used in the George scenario; and 3) representations of epistemic state and action effects used by planning. We describe each of the representations for each domain and how they capture uncertainty and knowledge gaps explicitly.

A. *Representations of Space*

Spatial knowledge is essential for a mobile robot, and many practical abilities depend on it, such as navigation, communication with humans about objects within a space, and the formation of episodic memories. In our system, spatial knowledge is represented in multiple layers, each at a different level of abstraction, from low-level sensory input to high-level conceptual symbols. Moreover, continuous space is discretized into a finite number of spatial units. The abstraction and discretization process is important as it makes the representation tractable and robust to changes in the world. Discretization also reduces the number of states considered, e.g., during the planning process.

The layered representation captures complex, cross-modal, spatial knowledge that is uncertain and dynamic. It assumes that knowledge should be captured only as accurately as is necessary to provide all necessary functionality. This reduces both the memory required, makes it robust to changes in the world, and reduces the effort required to synchronize the representation with the environment. In addition, uncertainties and gaps in spatial knowledge are explicitly modeled.

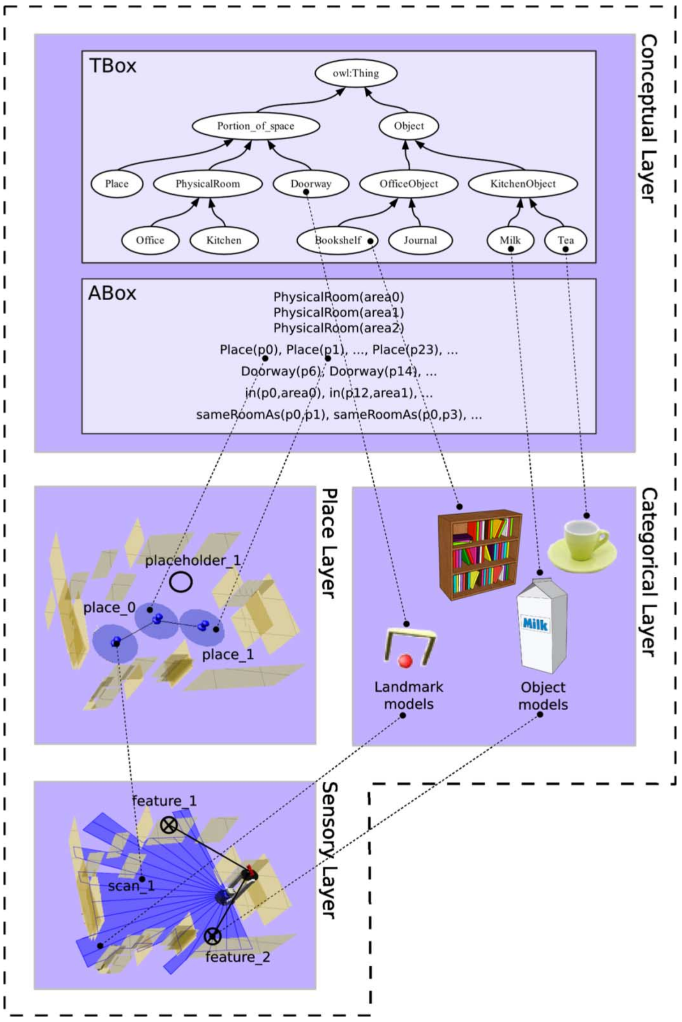


Fig. 4. Layered structure of the spatial representation. The position of each layer within the representation corresponds to the level of abstraction of the spatial knowledge. The ABox in the conceptual layer corresponds to the example in Fig. 1(b).

Fig. 4 gives an overview of the structure of the representation. It has four layers which model different aspects of the world, at different abstraction levels, and at different spatial scales. Each layer defines its own spatial entities and the way the agent's position in the world is represented. At the lowest abstraction level, we have the sensory layer which maintains an accurate representation of the robot's immediate environment extracted directly from the robot's sensory input. Higher, we have the place and categorical layers. The place layer provides fundamental discretization of the continuous space explored by the robot into a set of distinct places. The categorical layer focuses on low-level, long-term categorical models of the robot's sensory information. Finally, at the top, we have the conceptual layer, which associates human concepts (e.g., object or room category) with the categorical models in the categorical layer and groups places into human-compatible spatial segments such as rooms.

The following paragraphs provide additional details about each of the layers and their instantiations within our system. The system provides only an initial instantiation of the representation that validates correctness and usefulness of the knowledge structure within an integrated cognitive system. At the same time, as it will be mentioned below, some of the underlying algorithms do not adhere fully to the principles behind the representation. For a detailed theoretical discussion on those principles and optimal implementations, we refer the reader to [12].

1) Sensory Layer: In the sensory layer, a detailed model of the robot's immediate environment is created based on sensory input. It stores low-level features and landmarks extracted from the sensory input together with their exact position. Measures of uncertainty are also included in this representation. Landmarks beyond a certain distance are forgotten and replaced by new information. Thus, the representation in the sensory layer is akin to a sliding window with robocentric and up-to-date direct perceptual information.

The representation in the sensory layer helps to maintain stable and accurate information about the robot's relative movements. It also allows the robot to maintain and track the position of various features while they are nearby. Finally, the sensory layer provides the low level robotic movement systems with data for deriving basic control laws, e.g., for obstacle avoidance or visual servoing.

In the current implementation, the sensory layer is maintained by two subsystems: a metric SLAM algorithm [13] that builds a global metric map of the environment and an active visual search (AVS) component which represents hypotheses about objects found in the environment using a local grid map. The SLAM algorithm explicitly represents the uncertainty associated with the pose of the robot and the location of all landmarks using a multivariate Gaussian distribution encoded using a state vector and a covariance matrix [13], [14]. At the same time, the AVS component maintains hypotheses about the existence of an object of a specific category at a specific location using a probabilistic grid representation [15]. The probabilistic grid representation is formed from multiple cues about object location one of which is the presence of obstacles in the SLAM map. This is the prior on which basis the AVS algorithm determines the next best viewpoint based on a randomized art-gallery algorithm [16].

The existence of the global metric map violates some of the assumptions behind the proposed representation, however, it is only used internally. In future instantiations, the allocentric SLAM algorithm will be replaced by a robocentric method [17]–[19]. Here, in order to verify the correctness of such concept, we restrict access to the metric map from other components of the system, exposing only local and relative (with respect to the robot) metric information—with the exception of the navigation system that still uses the allocentric SLAM algorithm.

2) Place Layer: The place layer is responsible for the bottom–up discretization of continuous space. In it, the world is represented as a collection of spatial entities called *places* and their spatial relations. The aim of this representation is to represent the world at the level of accuracy sufficient for performing required actions and robust localization despite uncertainty and dynamic variations.

Besides places, the place layer also defines paths between them. The semantic significance of a path between two places is the possibility of moving directly between one and the other. In addition, the place layer explicitly represents *gaps in knowledge about explored space*. Space that has not yet been explored by the robot has no places in it. Therefore, hypothesized places are generated, which the robot would probably uncover if it moved in a certain direction. These hypothetical places allow for reasoning about unknown space, and for planning and executing exploratory activities. They are annotated as *placeholders* to distinguish them from actual places, but are otherwise identically represented and interconnected. For an illustrative example see Fig. 1(b).

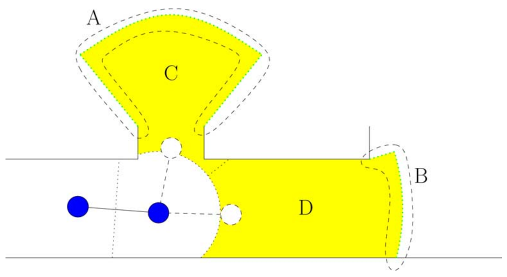


Fig. 5. Placeholder creation. Dashed circles are hypotheses, each representing one placeholder. A and B are frontier length estimates, C and D are coverage estimates for the respective placeholders.

Two quantitative measures are associated with each placeholder providing an estimate of information gain related to each exploration task. These are used by the goal generation and management system (Section V). The measures used are the *coverage estimate* (CE) and the *frontier length estimate* (FLE), cf. Fig. 5. The former is obtained by measuring the free space visible from the current node and not near to any existing node, and assigning it to the closest hypothesized place. This heuristically estimates the number of new places that would result from exploring in that direction. The FLE is the length of the border with unknown space. By prioritizing these two measures differently, different exploratory behaviors can be produced.

3) Categorical Layer: The categorical layer contains long-term, low-level representations of categorical models of the robot's sensory information, such as models of categories of objects like bookcases, phones, mugs, etc. These models of landmarks or objects are defined in terms of low-level features. The categorical models stored in this layer give rise to concepts utilized by higher-level layers. In many cases, complex models are required that can only be learned, not defined by hand. Models that correspond to categories understood by humans can be learned in a supervised fashion.

In our system, the categorical layer was realized through visual categorical models of objects employed by the AVS component and a simple door detection algorithm used as a landmark model. The AVS component uses the object recognition method proposed in [20] and the models associated with object classes reside in the categorical layer. However, using only this algorithm does not provide the pose of objects nor the uncertainty associated with it and is not robust to cases where two objects appear similar from a certain viewpoint. Therefore, a natural extension to this procedure which estimates the pose and class of objects is also implemented [15]. Additionally, in our experiments, we employed appearance and geometry-based models of place categories [21]. Although currently not being used in the

Dora scenario, those models constitute a part of the categorical layer.

4) Conceptual Layer: The conceptual layer provides a symbolic ontological representation of space that makes use of human-compatible concepts. The taxonomy of the spatial concepts and properties of spatial entities, as well as the instances of these concepts are linked to the lower levels of the spatial model. This associates semantic interpretations with the low-level models and can be used to specify which properties are meaningful, e.g., from the point of view of human–robot interaction. The main purpose of the conceptual layer is to represent a segmentation of the environment into rooms. Moreover it provides human-compatible concepts for these rooms based on the objects they contain, and it can supply default assumptions about which kinds of objects are likely to be found in which kinds of rooms.

The representation underlying the conceptual map is an OWL-DL ontology,[2] consisting of a taxonomy of concepts (*TBox*) and the knowledge about individuals in the domain (*ABox*), cf. Fig. 4, cf. [22]. Here is an example of a *concept definition* in the current implementation which defines a kitchen as a room that contains at least two typical objects

$$\mathsf{Kitchen} \equiv \mathsf{Room} \sqcap \geq 2\mathsf{contains.KitchenObject}.$$

Besides the usual inferences performed by the OWL-DL reasoner, namely *subsumption checking* for concepts in the TBox (i.e., establishing subclass/superclass relations between concepts) and *instance checking* for ABox members (i.e., inferring which concepts an individual instantiates), an additional *rule engine* is used to maintain a symbolic model of space under *incomplete* and *changing* information.

The discrete places from the place layer and their adjacency are the main pieces of knowledge that constitute the input for that reasoning. One, it maintains a representation that groups places into rooms. Furthermore, using observations (visually detected objects, appearance- and geometry-based room categories) it can infer human-compatible concepts for a room, and raise expectations about which other kinds of objects are prototypically likely to be present. The ongoing construction of the conceptual map is potentially nonmonotonic. The overall room organization may be revised on the basis of new observations. The further association between room concepts and salient, prototypical object types is established through the "locations" table of the OpenMind Indoor Common Sense[3] database by Honda Research Institute USA Inc.

In the current implementation, the conceptual layer can be used to determine *knowledge gaps in the categorization of rooms.* It is considered a gap in knowledge if for a given room (i.e., an instance of PhysicalRoom) its basic level category is unknown. This is assumed to be the case if no more specific concept than PhysicalRoom (i.e., Office or Kitchen, cf. Fig. 4) can be inferred for the individual. This knowledge gap persists until the robot has gathered enough evidence (i.e., contained objects) for inferring a subconcept.

[2]http://www.w3.org/TR/owl-guide/

[3]http://openmind.hri-us.com/

5) Space: Related Work: Multiple approaches to modeling and processing spatial knowledge have previously been published in the robotics and AI literature. Most of these approaches focus only on a fraction of spatial knowledge required by a mobile cognitive agent. The focus is either on building layered representations for the purpose of localization and navigation [23]–[27] or extraction of semantic information about the environment [21], [28], [29]. Meanwhile, several more comprehensive representations have been proposed. In [30], spatial and semantic information is represented hierarchically and links between the two hierarchies are established via anchoring. In [31], a multilayered representation consisting of a sensor, spatial, and cognitive layers is proposed for the purpose of human–robot interaction. Finally, [32] describes a hierarchical metric-topological-semantic representation consisting of places connected by doors and linked to semantic content encoded using objects and their properties. However, neither of those approaches provides an explicit representation of knowledge gaps. In contrast, in the proposed representation, hypotheses about object locations or unexplored space or gaps in the knowledge about the semantic category of rooms are explicitly modeled and used by the system for planning knowledge acquisition actions.

B. Representations of Epistemic State and Action Effects

Decisions about what actions to perform next are not preprogrammed in our robot, but are made by the *planning* subarchitecture. In this section, we describe how knowledge and knowledge-producing actions are modeled such that the planner can reason about how knowledge gaps can be filled. Planning systems traditionally use representations based on propositional logic. Most notably, the classic STRIPS formalism and its modern descendent PDDL are based on such a propositional representation. The representation we use for the planning system on our robot, however, is based on the SAS^+ formalism [33]. Here, instead of propositions, we use multivalued state variables (MVSVs) v, each with an associated domain $vdom(v)$ describing the set of possible values $x \in vdom(v)$ that v may assume. A state is a function s associating variables with values from their domain. In recent years, SAS^+ has been shown to enable powerful reasoning techniques in planning algorithms and systems based on SAS^+ now dominate the international planning competition. For the modeling needs of our robot applications, we have developed the SAS^+-based modeling language MAPL [34].

For robotic planning, MAPL provides, in addition to the computational advantages, several representational ones. First, it stays close to the feature/value model used by other subarchitectures of our robot. In particular, the mapping between binder states and planning states is greatly simplified: Roughly, each feature f of a union u in a belief model is mapped onto a state variable $f(u)$. For example, if the belief model describes that a room has been categorized as a kitchen by attributing the feature *areaclass : kitchen* to a union u, this would correspond to an assignment $areaclass(u) = kitchen$ in a planning state.

The main reason for using an SAS^+-based representation is that we can employ it to explicitly model knowledge and gaps in knowledge, so that the planner can efficiently reason about them. To this end, we must relax the assumption that in a state

s all variables v have a value x. Instead, we accept states that are only partially defined, i.e., where some variables are undefined or "unknown." Conversely, we also need to represent future states in which gaps will have been filled. By nature, we can not know in advance which value a variable v will assume then, but we can nevertheless exploit the knowledge that, e.g., after executing a sensing action the value of v will be "known." To this end, we use so-called *Kval* variables Kval_v with $vdom(\mathsf{Kval}_v) = \top, \bot$. With *Kval* variables we can also model the epistemic effects of sensing actions. For example, the action of running a room categorisation algorithm in a room is modeled in MAPL as follows:

```
(:sensor categorise_room
 :agent (?a - agent)
 :parameters (?r - room ?loc - place)
 :precondition
    (= (pos ?a) ?loc)
    (contains ?r ?loc)
 :effect (Kval ?a (areaclass ?r))
).
```

In words, this action model describes that an agent can sense the area class of a room, i.e., its being a kitchen, office or hallway, once the agent is at a place that belongs to the room in question. At planning time, the outcome of observing *areaclass(r)* is yet unknown, therefore the effect of $\texttt{categorise\_room}(\texttt{r}, \texttt{loc})$ is formally described as $\mathsf{Kval}_{\mathrm{areaclass(r)}} = \top$.

Of course, *Kval* variables can appear in goal formulae as well, so that we can conveniently express *epistemic goals*, i.e., goals concerned with closing knowledge gap. Goal formulae can contain expressions in first-order logic, in particular conditionals and quantifiers, so that we can give the robot goals like "categorize all rooms and explore all currently known places," which would correspond to $\forall loc.\mathsf{Kval}_{\mathrm{areaclass(loc)}} = \top \wedge \forall \mathrm{place.explored(place)}$.

Interestingly, due to the use of a quantified formula the goal will be reevaluated repeatedly during the continual planning process, i.e., the planner will autonomously adapt its plan to explore and categorize newly discovered places and rooms. A (slightly simplified) example of a plan using sensing actions that satisfy epistemic goals is given in Fig. 6.

In the George scenario and in our next instantiation of Dora, information will not only be obtained by sensing, but also through interaction with humans. To plan for such multiagent interactions, the robot must also reason about the knowledge of the other agents. We can express nested beliefs using MVSVs as well, e.g., "the robot R believes that human H believes that object o is a pen" is modeled as $\mathsf{B}^{R,H}_{\mathrm{type(o)}} = \mathrm{pen}$. Knowledge gaps may arise in several variants when nested beliefs are used, depending on which agent is ignorant of the other's belief. Again, with MVSVs we can represent the differences succinctly using agent-specific "unknown" symbols. Consider, e.g., the difference between the statements "R knows that H does not know the location of the cornflakes" ($\mathsf{Kval}^{R,H}_{\mathrm{pos(cornflakes)}} = \bot^H$) and "R does not know if H knows the location of the cornflakes" ($\mathsf{Kval}^{R,H}_{\mathrm{pos(cornflakes)}} = \bot^R$). Just as sensing actions are modeled using standard *Kval* variables, we can use nested *Kval* variables to describe *speech acts*. In particular, we can describe

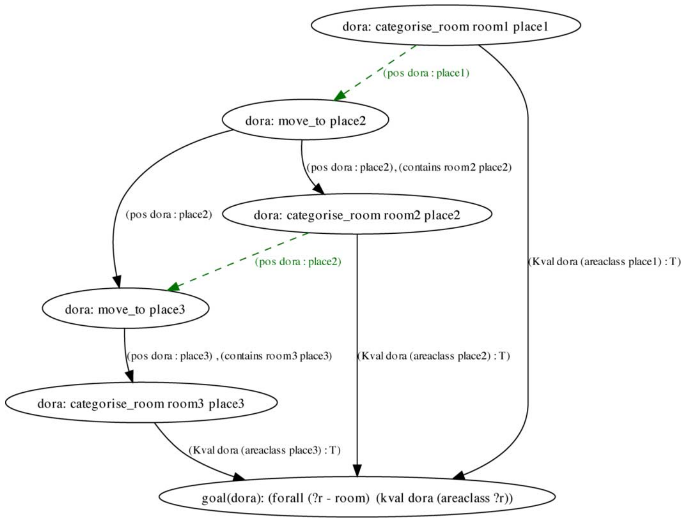

dora: categorise_room room1 place1
(pos dora : place1)
dora: move_to place2
(pos dora : place2), (contains room2 place2)
(pos dora : place2)
dora: categorise_room room2 place2
(pos dora : place2)
(Kval dora (areaclass place1) : T)
dora: move_to place3
(pos dora : place3) , (contains room3 place3)
(Kval dora (areaclass place2) : T)
dora: categorise_room room3 place3
(Kval dora (areaclass place3) : T)
goal(dora): (forall (?r - room) (kval dora (areaclass ?r))

Fig. 6. Plan using sensory actions to satisfy epistemic goals.

wh-questions and answers to them ("where," "what color," etc.) by modeling the appropriate nested belief effects. (Note: the planner was not used for dialog planning in the George system as presented in this paper, but will be in its next instantiation).

1) Planning: Related Work: Reasoning about knowledge is classically studied in the field of epistemic logic [35]. However, the representations developed there (e.g., possible-world semantics) are not directly usable in a planning system, because the required reasoning cannot be efficiently performed by a planner, which during the planning process must evaluate a number of belief states exponential in the size of the initial state description. The work presented here integrates concepts of epistemic logic into an expressive, yet suitable restricted planning representation so that a planning agent can efficiently reason about how it can change its own or another agent's state of knowledge in the future [34]. Similar approaches have been used previously to model planning in the presence of sensing [36], [37]. The MAPL representation presented here extends this to the multiagent case, i.e., it can additionally model and efficiently plan with nested and mutual beliefs as well as beliefs about knowledge gaps, all of which are crucial for reasoning about teamwork and dialog [38], [39].

C. Representations for Cross-Modal Learning

Cross-modal learning plays an important role in a self-extending system. It enables the system to, based on interaction with the environment and people, extend its current knowledge by learning about the relationships between symbols and features that arise from the interpretation of different modalities. It involves processing of information from multiple modalities, which have to be adequately represented. One modality may exploit information from another to update its current representations, or several modalities together may be used to form representations of a certain concept. In this subsection, we focus on the former type of interaction between modalities and present the representations that are used for continuous learning of basic visual concepts in a dialog with a human. While Section III describes the formation of belief models, which supervise the learning in the visual domain, this subsection focuses on representations that are being updated in this continuous learning process. All these principles are integrated and demonstrated in the George scenario described in Section VII.

1) Representations for Visual Concepts: The visual concepts are represented as generative models, probability density functions) over the feature space, and are constructed in online fashion from new observations. In particular, we apply the online kernel density estimator (oKDE) [40] to construct these models. The oKDE estimates the probability density functions by a mixture of Gaussians, is able to adapt using only a single data-point at a time, automatically adjusts its complexity and does not assume specific requirements on the target distribution. A particularly important feature of the oKDE is that is allows adaptation from the positive as well as negative examples [41]. The continuous learning proceeds by extracting visual data in a form of a high-dimensional features (e.g., multiple 1-D features relating to shape, texture, color, and intensity of the observed object) and oKDE is used to estimate the in this high-dimensional feature space. However, concepts such as *color red* reside only within lower dimensional subspace spanned only by features that relate to color (and not texture or shape). Therefore, during online learning, this subspace has to be identified to provide best performance. This is achieved by determining for a set of mutually exclusive concepts (e.g., colors green, blue, orange, etc.) the subspace which minimizes the overlap of the corresponding distributions. The overlap between the distributions is measured using the Hellinger distance as described in [42]. Therefore, during online operation, a multivariate generative model is continually maintained for each of the visual concepts and for mutually exclusive sets of concepts the feature subspace is continually being determined. The set of mutually exclusive concepts can then be used to construct a Bayesian classifier in the recognition phase, when the robot is generating a description of a particular object in terms of its color, shape, etc. However, since the system is operating in an online manner, the closed-world assumption can not be assumed; at every step the system should take into account also the probability of the "unknown model" as described in the following.

2) Accounting for Unknown Model: While maintaining good models of the visual concepts and being able to adapt those models is crucial for the robots online operation, the ability to detect gaps in the knowledge presented by these models is equally important. Generally speaking, the robot collects the visual information about its environment as follows. First, it determines a region in an image which contains the interesting information, then it "segments" that region and extracts the feature values z from which it later builds models of objects, concepts, etc. The visual information may be ambiguous by itself, and segmentation may not always be successful. We will assume that some measure of how well the segmentation was carried out exists and we will denote it by $s \in [0, 1]$. High values of s (around one) mean high confidence that a good observation z was obtained, while low values relate to low confidence.

Let $m \in \{m_k, m_u\}$ denote two possible events: i) the observation came from an existing internal model m_k; and ii) the observation came from an unknown model m_u. We define the knowledge model as a probability of observation z, given the confidence score s

$$p(z|s) = p(z|m_k, s)p(m_k|s) + p(z|m_u, s)p(m_u|s). \quad (1)$$

The function $p(z|m_k, s)$ is the probability of explaining z given that z comes from one of the learned models, $p(m_k|s)$ is the *a priori* probability of any learned model given the observer's score s. The function $p(z|m_u, s)$ is the probability of z corresponding to the unknown model, and $p(m_u|s)$ is the probability of the model "unknown" given the score s.

Assume that the robot has learned K separate alternative internal models $M = \{M_i\}_{i=1:K}$ from previous observations. The probability $p(z|m_k, s)$ can then be further decomposed in terms of these K models

$$p(z|m_k, s) = \sum_{i=1}^{K} p(z|M_i, m_k, s)p(M_i|m_k, s). \quad (2)$$

If we define the "unknown" model by M_0 and set $p(z|m_u, s) = p(z|M_0, m_u, s)p(M_0|m_u, s)$, then (1) becomes

$$p(z|s) = p(m_k|s) \sum_{i=1}^{K} p(z|M_i, m_k, s)p(M_i|m_k, s) + p(m_u|s)p(z|M_0, m_u, s)p(M_0|m_u, s). \quad (3)$$

Note that the "unknown model", M_0, accounts for a poor classification, by which we mean that none of the learned models supports the observation z strongly enough. We assume that the probability of this event is uniformly distributed over the feature space, which means that we can define the likelihood of model M_0, given observation z by a uniform distribution, i.e., $p(z|M_0, m_u, s) = \mathcal{U}(z)$. Note also, that the only possible unknown model comes from the class M_0, therefore $p(M_0|m_u, s) = 1$.

The observation z can be classified into the class M_i which maximizes the a posteriori probability (AP). The a posteriori probability of a class M_i is calculated as

$$p(M_i|z, s) = \frac{p(z|M_i, m, s)p(M_i|m, s)p(m|s)}{p(z|s)} \quad (4)$$

where $m = m_k$ for $i \in [1, K]$ and $m = m_u$ for $i = 0$.

In our implementations, the distribution of each ith alternative of the known model $p(z|M_i, m_k, s)$ is continuously updated by the oKDE [40], while the *a priori* probability $p(M_i|m_k, s)$ for each model is calculated from the frequency at which each of the alternative classes M_i, $i > 0$, has been observed. The *a priori* probability of an unknown model (and implicitly of a known model), $p(m_u|s)$ is assumed nonstationary in that it changes with time. The following function decreases the "unknown" class probability with increasing number of observations N:

$$p(m_u|s) = e^{-0.5(N/\sigma_N)^2} \quad (5)$$

where σ_N is a user specified parameter that specifies how the robot's internal confidence about learned models changes with time.

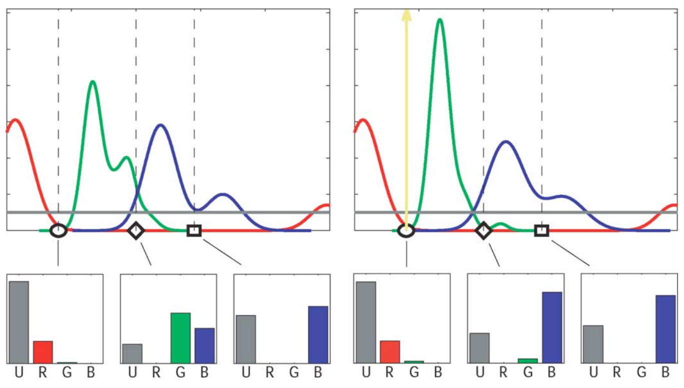


Fig. 7. Example of detecting the knowledge gaps and updating the 1-D KDE representations. Top row: probability distributions for three colors (red, green, and blue) and unknown model (the horizontal gray line) in 1-D feature space. Bottom row: *a posteriori* probabilities for the unknown model (U) and three colors (R, G, B) for three feature values denoted by the circle, the diamond, and the square. Left column: before updates, right column: after updates.

With above definitions, the knowledge model is completely defined and allows discovery of knowledge gaps. They can be discovered through inspection of the AP distribution. In particular, we can distinguish two general cases:

- the observation z can be best explained by the unkown model, which indicates the gap in the knowledge; the observation should most probably be modeled with a model, which has not yet been learned;
- the *a priori* probability of the model that best explains the observation is low, which indicates that the classification is very uncertain and that the current model can not provide a reliable result.

3) Illustrative Example: For a better visualization of the knowledge update and gap discovery, we will restrict our example to a one-dimensional case. Fig. 7 illustrates detection and filling of knowledge gaps for three cases (feature values) denoted by the circle, the diamond, and the square. The plots in the left column depict the models and the recognition at a particular step in the learning process, while the right column depicts the situation after the system has updated these models considering the detected knowledge gaps and the answers from the tutor.

Consider a scene similar to that presented in Fig. 1. Let us assume that the circle in Fig. 7 represents the yellow object and that the yellow color has not been presented to the robot before. Therefore, the corresponding model for color yellow has not yet been learned and the feature value obtained from the segmented yellow object fails in a not yet modeled area. This value is thus best explained by the "unknown model," which has the highest a posteriori probability. The robot detects this gap in his knowledge and asks the tutor "*Which color is this object?*," and after the tutor provides the requested information, the robot initializes a model for yellow color. However, since only one sample does not suffice to build a reliable representation, the yellow color will only be able to be recognized after some additional yellow objects are observed.

The feature value denoted by a diamond in Fig. 7 is best explained by a green model, however, this recognition is not very reliable, therefore the robot asks the tutor, "*Is this object green?*" to verify its belief. After the tutor replies, "*No. It is blue.*," the robot first unlearns the representation of green and updates the representation of blue. The corrected representations, depicted in the right column in Fig. 7, then enable the correct recognition as indicated by the second bar plot in the right column of the Fig. 7.

The last case denoted by the square shows another example of nonreliable recognition, which triggers the additional clarification question to the tutor: "*Is this object blue?*." After the robot gets a positive answer, it updates the representation of blue, which increases the probability of the recognition.

4) Cross-Modal Learning: Related Work: The central topic of cross-modal learning as addressed in this work is continuous estimation of classifiers of concepts from streaming data. The related methods [43]–[46] aim at generating these classifiers from batches of data and require the user to provide labels for the subsets of the batches. In real-world scenarios in which a robot interacts with a tutor, it is often desirable both to allow on-the-fly learning, which requires learning from a single sample at a time, and also the detection of knowledge gaps on-the-fly. The main reason for choosing the online kernel density estimator (oKDE) [40] was its ability to construct probabilistic generative models that can be adapted using only a single data-point at a time that can easily be turned into classifiers that account for the unknown classes as well.

V. Goal Management: Choosing Between Different Epistemic Goals

Previous sections have focused on *representation* of knowledge gaps and uncertainty. We now describe a *goal generation and management framework (GGM)* that enables the robot to decide *which* gaps or uncertainties to eliminate *when*. To do this the robot has to analyze representations residing in each subarchitecture, monitor for gaps and uncertainties, quickly pick an appropriate subset to be filled, and translate those into goal statements for the planner (see Section IV-B). This must be done in a way that is: 1) *generic* to cope with different types of knowledge gaps and uncertainties; 2) *scalable* as the number of gaps in as system rises; and 3) *informed* with regard to the robot's overall task and the relevance of the different possible epistemic goals to that task.

We have built on the work of [47], to produce the design illustrated in Fig. 8. This design is a general framework, or schema, for an architecture for goal generation and management that tackles the mentioned issues. It specifies a collection of interacting elements which must be included in any instantiation of the framework, although the precise details of the instantiation will inevitably vary between instances. The elements of the framework are described in more detail below.

At the bottom of the framework, a system's drives are encoded as multiple *goal generators*. These are concurrently active processes which monitor the system's state (both the external world and internal representations) and produce *goals* to satisfy the system's drives. Generators can also remove previously generated goals if they are judged to no longer be appropriate. In this manner, we can say that the system's drives are encoded in the goal generators (either explicitly or implicitly). We work from the assumption that as a goal passes up through the

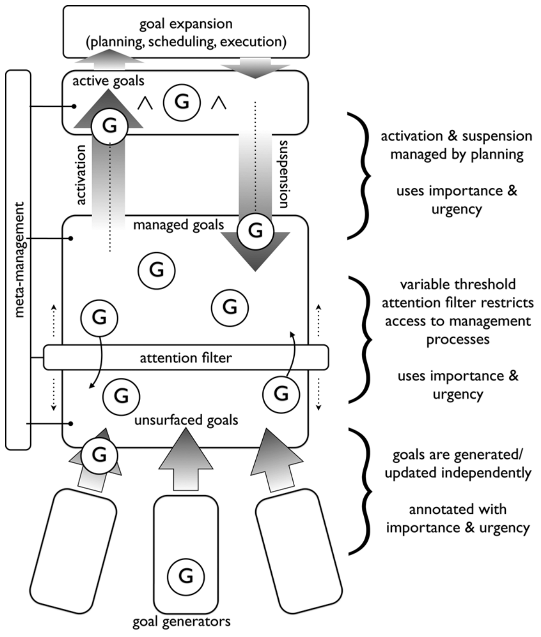

goal expansion
(planning, scheduling, execution)
active goals
meta-management
activation
suspension
managed goals
attention filter
unsurfaced goals
goal generators
activation & suspension managed by planning
uses importance & urgency
variable threshold attention filter restricts access to management processes
uses importance & urgency
goals are generated/ updated independently
annotated with importance & urgency

Fig. 8. Goal generation and management framework.

framework from a generator and influences a system's behavior, it is inspected by processes of greater and greater computational complexity. Therefore, the lower strata of the framework exist to protect these processes (and thus overall system resources) from having to consider more goals than is necessary (where this could be a contingent judgement). The main mechanism in the framework for protecting the management processes is the *attention filter*. This is a coarse barrier which uses simple, fast processing to let some goals through to the management mechanisms whilst blocking others. Goals which make it through this filter are described as *surfaced*, thus the goals which fail to pass the filter are referred to as *unsurfaced*. A collection of management processes determine which of the surfaced goals should be combined to serve as the goals being actively pursued by the system. If a goal is selected in this way, we describe it as *activated*. If a goal is removed from the set of goals being pursued by the system, we refer to it as *suspended*.

In order to fulfil their roles, the filtering and management processes require information on which to base their decisions. Following the original work [47], the framework requires that goal generators annotate each goal with a description of the goal's *importance* and *urgency*, and keep these descriptions up to date as long as the goal exists. Importance should reflect the significance of the goal to the agent (as motivated by the related drive). Urgency should reflect the necessity of achieving the goal sooner rather than later. As we will see later, producing importance and urgency descriptions for use in such a framework is a problem in itself. In addition to these descriptions, the framework allows the management processes to use whatever approaches are required to select and maintain a set of active goals. Perhaps the minimum requirements on these processes is the ability to check whether a goal, or collection of goals, can be achieved (thus, positing planning as a goal activation, as well as achievement, mechanism).

The GGM approach is currently implemented as one of the core concepts in our exploring robot Dora (cf. Section VI). In this system, we derive the importance of a goal from an estimated *information gain* computed for the epistemic goals. In brief, the information gain for achieving the goal of exploring a yet unexplored place is derived from the measures shown in Fig. 5. The information gain of categorizing a room is similarly designed, assuming that a categorizing bigger rooms yields more information. The GGM continuously monitors these measures of information gain and relates it to the *costs* to actually achieve this goal acquired by asking the planner. We are currently not employing a notion of *urgency* in our implementation as the robot's drives are not prioritized so far.

GGM in cooperation with planning implements *action selection and execution* in our systems, allowing the robots to expose effective and efficient self-extending behavior.

A. Goal Management: Related Work

Our systems have *curiosity* to extend their knowledge and decrease uncertainty, and to this extent our work is comparable to that of Oudeyer *et al.* [48] on *intelligent adaptive curiosity (IAC)*. In contrast to their approach, which focuses on specific learning mechanisms applied to motor skills in a developmental way, we emphasize representations and methods for reasoning with them. To cope with the complexity introduced by this approach we follow a widely used decomposition of the problem into goal generation, goal management, and planning [49]. Goal generation in our approach can be seen as an implementation of the notion of a desire from BDI [50]: the robot generates goals to accommodate the desire to extend knowledge. Our goal management based on *filtering* is inspired by mechanisms of opportunism and prioritization [51]. The need for reducing the set of goals to be planned for has also been highlighted in the work of Coddington *et al.* [52]. Goal management could be tackled by a planning algorithm capable of oversubscription planning [53], which can find the optimal subset of achievable goals. The primary difficulty with this is obtaining reliable models, and how to plan with limited resources. Schermerhorn *et al.* [54] highlight further difficulties of treating goal management as a rational decision making problem. These difficulties are primarily why we employ the predicted information gain to heuristically select the goals to be planned for.

VI. Dora the Explorer

The Dora scenario focuses on spatial representations and two types of knowledge gaps that give rise to the two epistemic goals *explore a place* and *categorize a room*. Dora's system architecture (Fig. 9) is composed of five of the subarchitectures discussed earlier, all running onboard the robot [Fig. 1(a)]. Fig. 9 is adopted from the UML 2.0 specification and illustrates information flow in the system. Most of the flow goes via the working memories in an event-driven manner as described in Section II, but for readability Fig. 9 instead shows sources and sinks of information linked directly, following the shorthand of Fig. 2(b). Neither does Fig. 9 include all components, focusing instead just on those required to understand the way architecture enables self-understanding and self-extension. Dora is different from George in several ways: using spatial.sa extensively and

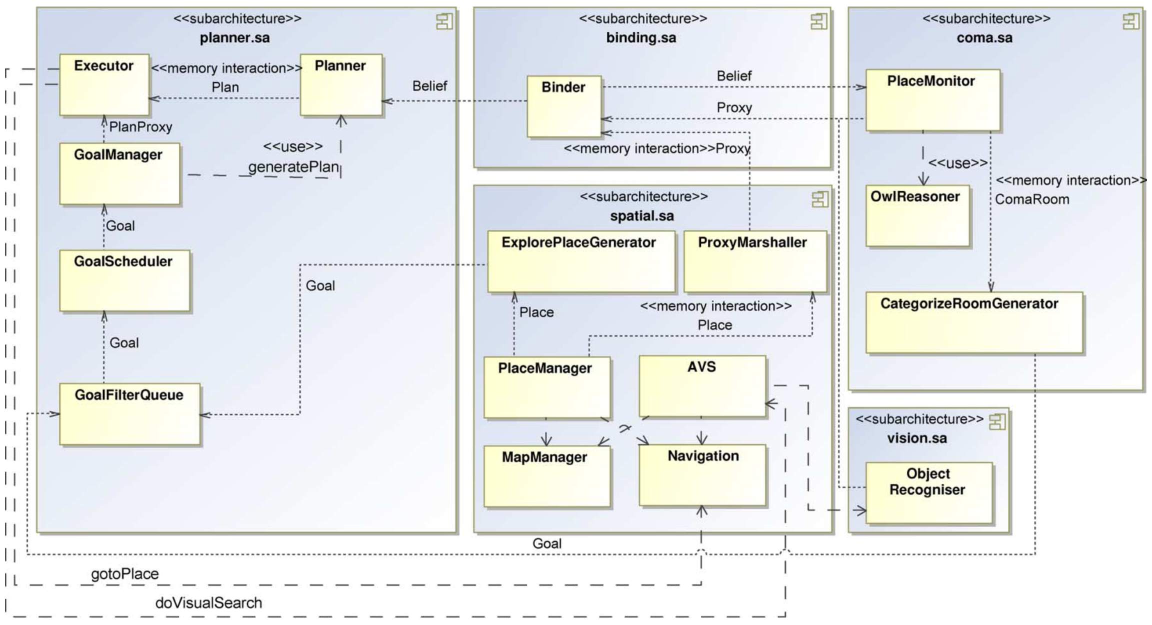


Fig. 9. Dora system architecture. For clarity, all memory interactions are not depicted as information flow mediated through working memories in the subarchitectures but as directed dotted connections of sources and sinks. The dashed lines represent synchronous request–reply calls to components by mutually modifying memory structures.

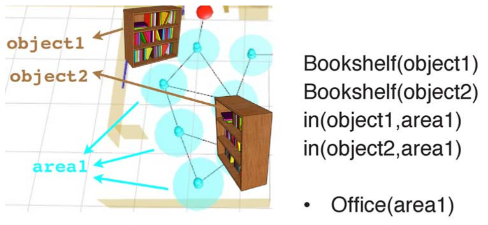
also the system for reasoning about spatial concepts (coma.sa). However vision.sa only plays a minor role in this system. The ObjectRecognizer component (based on FERNS [55]) detects objects when triggered by AVS. The ProxyMarshalling component selects spatial information to be presented as a Proxy to the binder. The PlaceMonitor in coma.sa fulfills a similar purpose. As described previously, binding.sa builds a unified representation of beliefs and proxies, to provide the information required by the Planner. Goal management in Dora is implemented as part of planner.sa. Epistemic goals are created by identifying knowledge gaps in the representations in the working memories of other subarchitectures as discussed in Sections IV-A2 and IV-A4. These goals are filtered, scheduled, and planned for as described in Section V. The Executor component executes and monitors actions by triggering either the Navigation component to explore a place or the AVS component (cf. Section IV-A1) to plan and execute visual search for objects that allow the OwlReasoner to draw inferences about rooms (cf. Section IV-A4).

Fig. 9 also shows how the decomposition into subarchitectures eliminates unnecessary information flow. The only representations exchanged between subarchitectures are those related to binding (cf. Section III) and the epistemic goals corresponding to the knowledge gaps.

A. Dora Running

A prototypical run with the current Dora implementation unfolds as follows:

1) Dora starts from scratch without any knowledge about the specific environment she is operating in.
2) Optionally, Dora can be given a short tour by a human instructor to create an initial map already containing some knowledge gaps as shown in Fig. 1(b).
3) Dora autonomously explores her environment having drives to fill two types of gaps in spatial knowledge: unexplored places as defined in the place layer and uncategorized rooms as defined in conceptual layer. In the example in Fig. 1(b), the rooms area0, area1, and area2 give rise to room categorization drives, whereas the different placeholders lead to exploration goals. Note that a number of placeholders (notably the ones labelled "8(7)," "7(16)," and "20(19)") are in space that will later be segmented into new rooms, which will in turn need to be categorized.
4) A room categorization goal is considered satisfied when a more specific concept can be inferred for a PhysicalRoom instance in the ABox of the conceptual map layer, cf. Fig. 10. An exploration goal is satisfied if the robot either turns the placeholder into a real place or discards it as a false hypothesis.



(a) (b)

Fig. 10. Continuing the example in Fig. 1(b): Based on the presence of two OfficeObject instances the DL reasoner infers that area1 instantiates Office. (a) Dora has found two bookshelves. (b) New facts stored in the ABox.

The two types of gaps are created and monitored by the components PlaceManager@spatial.sa and PlaceMonitor@coma.sa, respectively. Fig. 9 illustrates how these components submit hypotheses about ComaRoom (a detected but not yet categorized room) and Place (a detected but not yet explored place) to their working memories. From these gaps, epistemic goals are generated by the goal generators

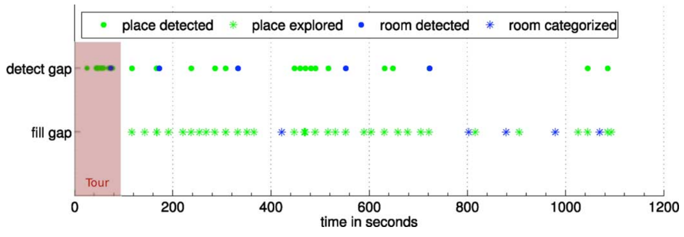


Fig. 11. Exemplary course of action for filling knowledge gaps in a real run.

ExplorePlaceGenerator and CategorizeRoomGenerator, respectively. Thus, several goals are always present in the system corresponding to these gaps. Dora's behavior is driven by a subset of these goals as selected by GGM, cf. Section V) and the resulting plans to achieve these goals.

Fig. 11 shows for an example run the rate at which Dora detects and fills the two types of knowledge gaps. The x-axis shows the time in seconds since the beginning of the run. This particular run comprised an initial tour taking Dora from the corridor [the long room in the center of Fig. 1(b)] to the room in the upper-right corner in that figure. It can be seen in Fig. 11 that she is passively detecting gaps in her knowledge in the phase labelled "Tour," but not yet autonomously extending it. After the tour Dora interleaves categorization of rooms with exploration of new places. A video[4] of the real robot operating is available to aid understanding.

Taking a closer look at the actual processes in the system, we can see how the interactions between components work. Fig. 12 pictures the activity that the robot goes through from detecting to filling a knowledge gap. This corresponds to "explore place" only, but the structure is similar for "categorize room." It starts with spatial.sa hypothesizing a new place and thus, generating a Place in the working memory that is marked as being hypothetical. This generation triggers binding and ExplorePlaceGenerator to create a Belief about this place and an epistemic Goal to explore this place. After the motivation-related components have filtered and scheduled the generated goal, the planner is triggered to generate a plan to actually achieve it. The Executor then executes the actions of the plan. One action will be to navigate towards the placeholder which will, in this example, make it explored. This update is again propagated through the working memory, resulting in the goal being removed and the belief being updated asynchronously.

B. Dora: Experimental Results

Our evaluation of Dora comprises runs in both real and simulated environments. The first is a proof of our approach in the real world. The second enables the high reproducibility and greater experimental control necessary for entirely fair analysis of system performance for different treatments. Our simulation framework only substitutes the sensors and actuators, the core system runs unmodified. Because of this runs in simulation take roughly as much time to complete as real runs. The simulations used the floor plan of one of the real environments in which Dora operates (cf. Fig. 13). We first briefly report some results of experiments conducted in a real flat with Dora, before detailing two experiments carried out in simulation. The first of those

[4]http://cogx.eu/results/

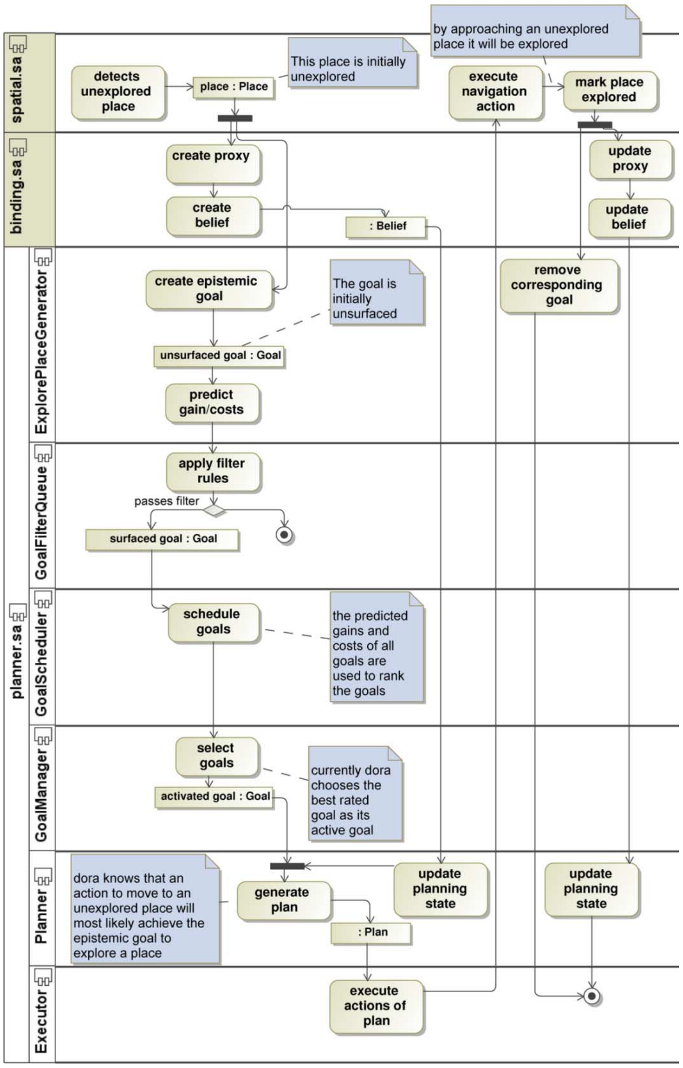


Fig. 12. Activity diagram illustrating the "path" of a knowledge gap from its generation to its filling.

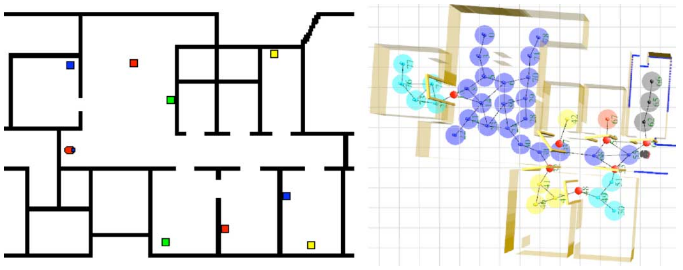


Fig. 13. Stage simulation model used in the experiments (l) and screenshots of the visualization tool acquired during one of the three experiments (r).

examines the quality of the spatial representations with an emphasis on the nonmonotonic reasoning in the conceptual layer. The second simulation experiment studies the benefits of the goal management framework.

1) Experiment 1: Real World Runs: The real environment we tested Dora in was composed of a kitchen, a corridor, and two bedrooms. Each bedroom contained between three and four objects that Dora can recognize and use for the categorization of that room. The kitchen, being larger, contained 6 objects. The robot was given a short tour without any categorization of the

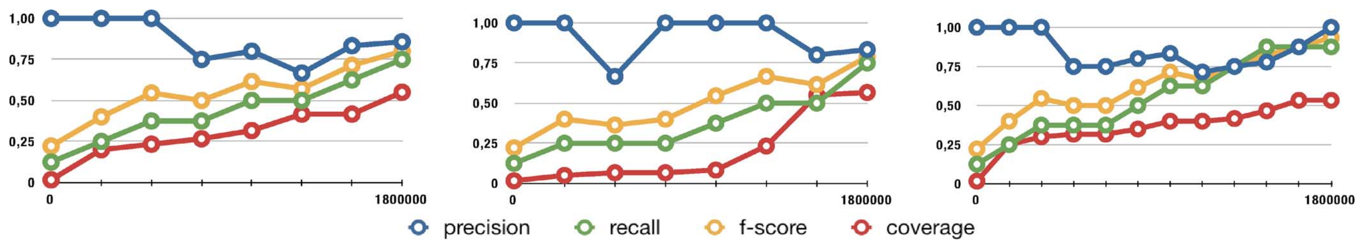


Fig. 14. Plots for precision, recall, balanced f-score, and coverage of each of the three experimental runs. The Y-axis shows the normalized values for precision, recall, balanced f-score, and coverage (0–1). The X-axis is time, in ms.

rooms by a human through the corridor one of the bedrooms and part of the kitchen, and then left to explore autonomously. Dora succeeded in 18 out of 21 runs to explore all the space and categorize all rooms correctly. In average, it took Dora 26 min to accomplish the full task in these runs; of which she spent 11.45 s planning for goals selected by the goal management scheme. The average total distance Dora traveled in a run was 104 m. Without discussion of the details here, the most prominent failures in the remaining three runs were caused by failures to detect the second bed room due to self-localization errors, or on detection of spurious or inaccurately located doors in the mapping process.

2) Experiment 2: Spatial Representation: One consequence of the uncertainty and partiality of the observations Dora makes is that the map building process is nonmonotonic. Conclusions may need to be reconsidered in the light of new evidence. In this experiment, we assess the accuracy of the nonmonotonically built spatial representation as it evolves through a robot run.

Setup: The map consisted of eight rooms: a corridor, a terminal room, a lab, two offices, two restrooms, and a printer room, cf. Fig. 13. This constitutes the ground truth for our tests of the accuracy of the room maintenance. The robot was ordered to perform an autonomous exploration, which means that only placeholder exploration goals were considered by the motivation system. To evaluate the coverage that this exploration yields, we determined a gold standard of 60 place nodes to be generated in order to fully, densely and optimally cover the simulated environment. We achieved this by manually steering the robot to yield an optimal coverage, staying close to walls and moving in narrow, parallel lanes.

We performed three runs with the robot in different starting positions, each time with an empty map. Each run was cut-off after 30 min. The robot was then manually controlled to take the shortest route back to its starting position.

For the evaluation, the system logged the state of its ABox each time a new room was created, or an existing one was deleted. This subsumes cases in which rooms are split or merged. At each such step, the generated map was compared to the ground truth for the room representation and to the gold standard for Place node coverage. The first room instance to cover part of a ground-truth room is counted as *true positive (TP)*. If that room instance extends into a second room, it is counted as TP only once, and once as a *false positive (FP)*. Each additional room instance inside a ground-truth room is also counted as FP. *False negatives (FN)* are ground-truth rooms for which no room instance exists. Using these measures, precision P, recall R and the balanced f-score F for the room maintenance are as follows: $P = |TP|/(|TP|+|FP|)$, $R = |TP|/(|TP|+|FN|)$, $F = 2 \times ((P \times R)/(P + R))$. We compute a normalized value for coverage using coverage $= |\text{nodes}|/60$.

Results: Fig. 14 shows the development of the relevant measures during the three experimental runs. As can be seen, the accuracy (balanced f-score) of the representation monotonically increases towards a high end result (0.8, 0.79, and 0.93, resp.). The increases and decreases in precision during the individual runs are due to the introduction and retraction of false room instances. Recall can be interpreted as coverage in terms of room instances. After 30 min, the exploration algorithm yielded a relatively high recall value (0.75, 0.75, and 0.875, respectively), i.e., most of the rooms had been visited. A recurring problem here was that the two smallest rooms were often only entered by a few dm. This was enough to consider the corresponding placeholder to be explored, but not enough to create an additional place node beyond the doorway, which would have been the prerequisite for room instance creation. The node coverage that the algorithm achieved after 30 min (33, 34, 32, out of 60, respectively) can be attributed partly to the 30-min cut-off of the experiment, and partly to the exploration strategy which goes for high information gain placeholders first. These tend to be in the middle of a room rather than close to its walls.

3) Experiment 3: Goal Management: As discussed in Sections IV-B and V, there are two possible ways to express Dora's drives. First, we can explicitly use quantifiers to create a single, persistent, conjunctive goal to explore all places and categorize all rooms and rely on the replanning ability of continual planning (cf. Section IV-B) to replan for the same goal as new hypothesized places arise. We term this system setup the *conjunct goal set (CGS)*. The alternative is to use the goal generation and management approach to repeatedly select from the possible goals a subset for planning. Our hypotheses are that: 1) the effort for planning is reduced as we chunk the problem into smaller pieces, making it tractable for more complex problems; and 2) goal management is a powerful and simple means to encode domain knowledge into the behavior generation in Dora. We refer to this second setup as the *managed goal set (MGS)*. Overall, the goals in both sets are the same, just the process of selection is different.

Setup: For this study we restricted the space to be explored to five rooms and the corridor [being the right part of Fig. 1(b) without the large room]. The goal is to categorize all rooms by using ten objects, placed two per room in each of the five rooms, to draw inferences about room category. Each object is associated with one of three categories in the OpenMind indoor

TABLE I
PLANNING TIME MEASURES (ALL IN S)

	CGS	**MGS**
avg. time per planning call	0.621 s	0.292 s
avg. time spent on planning	48.843 s	8.858 s

common sense database (room, office, and kitchen). A run starts with a short tour through the corridor, followed by autonomous operation. Fig. 11 is generated from one of these runs including the tour and the categorization of five rooms. In total, we ran the system 15 times: eight in MGS configuration and seven in CGS. A run for the CGS configuration completed when the conjunctive goal was achieved (i.e., no places left unexplored and no rooms uncategorized). The MGS configuration completed when no more surfaced goals remained.

Results: As part of our experiments (detailed in [56]), we were interested in the effect the MGS approach has on planning time, and so measured this under the MGS and CGS configurations (Table I). All the differences shown are statistically significant with $p < 0.0001$ using a Mann–Whitney test. As the first row of the table indicates, there is a significant difference between the average time taken by a single call to the planner. A call occurs either when the goal management activates a new goal or when replanning is triggered by a state change. Planning calls in CGS take more than twice the time compared to MGS. This is due to the higher complexity of the planning problems in the CGS configuration (it is planning for the conjunction of all epistemic goals rather than a single goal). If we look at the average time spent on planning in total per run (second row in Table I), the difference is more prominent. This is due to the fact that in the CGS configuration the planner is triggered more often: 79.0 times on average, compared to 31.1 times for the MGS configuration. This is because the longer plan lengths required in CGS are more likely to be affected by state changes and thus require more frequent replanning.

Fig. 15 shows how the complexity of planning problems evolves as the system is running. It depicts the length of single planner calls against the runtime of the system. For comparability, this plot has been created from a subset of the completely successful runs (five of each configuration). The planning time is averaged at discrete time steps across all the runs of each setup. From this figure and Table I, it is clear that less planning effort is required in MGS compared to CGS, and that the trend is steeper for CGS than for MGS. This is caused by the fact that in CGS Dora has to repeatedly plan for all possible goals over a continually growing map. MGS planning time also grows, but more slowly. This supports our hypothesis that with a suitable mechanism for goal selection we can scale our approach to increasingly complex environments and with high numbers of knowledge gaps in our representations.

VII. GEORGE: CURIOSITY-DRIVEN CROSS-MODAL LEARNING

The George scenario is concerned with *learning the association between visual features of an object and its linguistically expressed properties*. It involves a conversation between the robot and a human tutor [see Fig. 1(c)]. The robot is asked to recognize and describe the objects in a table top scene, of which there are up to five. The human can move or remove objects from the table during the dialog, and teach the robot about the objects by describing their properties. Initially, the tutor drives the learning, but after a while, the robot takes the initiative, and is able to learn either without verbal feedback, or by asking the tutor for clarification when necessary. To achieve this the robot must establish a common understanding with the human about what is in the scene, and verbalize both its knowledge and knowledge gaps. To test what the robot has learned the tutor asks questions about the scene. The goal of learning is for the robot's representations to be rich enough to correctly describe the scene.

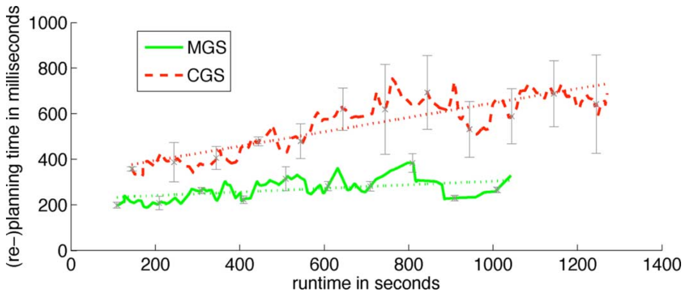


Fig. 15. Averaged planning time during a system run.

Two types of learning are present in the George system, which differ in terms of the source of the motivation to learn. In tutor driven learning the learning process is initiated by the human teacher, while in tutor assisted learning, the learning is triggered by the robot.

Tutor-driven learning is suitable during the initial stages, when the robot has to be given information to reliably initiate its visual concepts. Consider a scene with a single object present.

A: Do you know what this is?
B: No.
C: This is a red object.
D: Let me see. OK.

After George gets this information, he can initiate his visual representation of redness. After several such learning steps, the acquired models become reliable enough that they can be used by George to refer to individual objects, and to understand references by the human. From this point on there can be several objects in the scene at the same time, as in Fig. 1, and George can answer questions about some of them.

A: What color is the elongated object?
B: It is yellow.

When enough of the models are reliable, George can take the initiative and drive the learning by asking questions of the tutor. He will typically do this when he is able to detect an object in the scene, but is not certain about some or all of its properties. As described in Section IV-C, in such *tutor assisted* learning there are two types of uncertainty and gaps. If the object does not fit any previously learned models, he considers there to be a gap in his knowledge and asks the tutor to provide information about its novel property.

A. Which color is this object?
B. It is yellow.
C. OK.

The robot is now able to initialize the model for yellow and, after the robot observes a few additional yellow objects, which

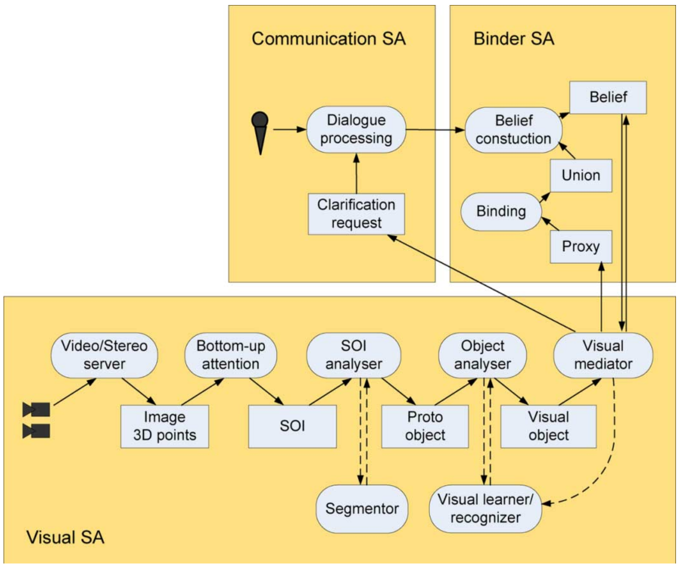


Fig. 16. Architecture of the George system.

make the model of yellow reliable enough, he will be able to recognize the yellow color.

In the second case, the robot is able to associate the object with a particular model, however, the recognition is not very reliable. Therefore, the robot asks the tutor for clarification.

A. Is this red?
B. No. This is yellow.
C. OK.

After the robot receives the answer from the tutor, he corrects (unlearns) the representation of the concept of red and updates the representation of yellow. In such mixed initiative dialog, George continuously improves his representations of basic visual concepts. After a while, George can successfully recognize all the acquired concepts to provide reliable answers to a variety of questions.

A: Do you know what this is?
B: It is a blue object.
C: What shape is the red object?
D: It is elongated.

A. System Architecture and Processing Pipeline

The George system is composed of three subarchitectures (SAs): the *binder SA*, *communication SA*, and *visual SA* [57] (see Fig. 16). The components of visual SA can be divided into three layers: the quantitative layer, the qualitative layer, and the mediative layer [58].

The *quantitative layer* processes the visual scene as a whole and implements a bottom–up attention mechanism to identify regions in the scene for further visual processing. This uses a stereo 3-D point cloud provided by the *stereo reconstruction component* to extract the dominant horizontal planes and the things sticking out from those planes. Those sticking-out parts form spherical 3-D spaces of interest (SOIs). The *SOI analyzer* component validates the SOIs and, if deemed interesting (based on SOI persistence, stability, size, etc.), upgrades them to *proto-objects* adding information that will be needed by the qualitative layer (e.g., a segmentation mask).

The *qualitative layer* processes each interesting proto-object individually, focusing on qualitative properties. After the extraction of visual attributes (by the visual learner–recognizer), like color and shape, the *object analyzer* upgrades the proto-objects to *visual objects*. Visual objects encapsulate all the information available within the visual SA and are the final modal representations of the perceived entities in the scene. Also, the learning of visual attributes is performed on this layer.

The *mediative layer* exchanges information with other modalities. This is done via the the binder SA (Section III). For each visual object, the *visual mediator component* creates a proxy in the binder. This component also monitors beliefs for possible learning opportunities, which result in modality specific learning actions. Another important function of the mediator is to formulate and forward clarification motivations in the case of missing or ambiguous modal information. Currently, these motivations are intercepted by the communication SA, which synthesizes a question about the uncertain object property.

We now give an example of this processing pipeline, where the human has just placed several objects in the scene (see Fig. 1) and refers to the only elongated object in the scene (the yellow tea box) by asserting "*H: The elongated object is yellow*."

At this point in visual SA, the tea box is represented by a *SOI* on the quantitative layer, a *proto-object* on the qualitative layer, and a *visual object* on the mediative layer. Let us assume that the *visual learner–recognizer* has recognized the object's elongated shape, but has failed to recognize its color. In the binder, this results in a one-proxy union with the binding features giving the highest probability to the elongated shape, while the color is considered unknown. This union is referenced by a private belief in the belief model (Fig. 17, step 1).

The tutor's utterance "The elongated object is yellow" is then processed by the communication SA, resulting in a new belief *attributed to the tutor*. This attributed belief restricts the shape to elongated and asserts the color to be yellow. Before this belief is actually added to the belief model, the binder translates it to a binding proxy (phantom proxy) with the shape restriction as a binding feature. In the most probable configuration, the phantom proxy is bound to the existing union, which already includes the visual proxy representing the tea box (Fig. 17, step 2). The union is promptly referenced by the attributed belief and the phantom proxy is deleted soon after.

In visual SA, the mediator intercepts the event of adding the attributed belief. The color assertion and the absence of the color restriction in the robot's belief is deemed as a learning opportunity (the mediator knows that both beliefs reference the same binding union, hence the same object). The mediator translates the asserted color information to an equivalent modal color label and compiles a learning task. The learner–recognizer uses the label and the lower-level visual features of the tea box to update its yellow color model. After the learning task is complete, the mediator verifies the attributed belief, which changes its epistemic status to shared (Fig. 17, step 3). The learning action retriggers the recognition. If the updated yellow color model is good enough, the color information in the binder and belief model is updated (Fig. 17, step 4).

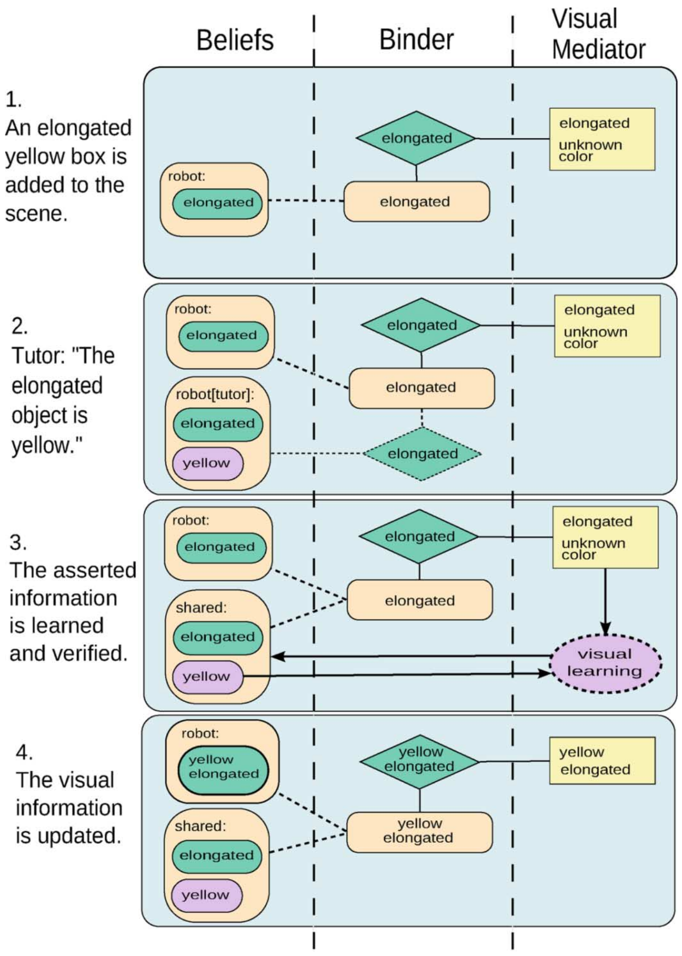

Beliefs
Binder
Visual Mediator
1. An elongated yellow box is added to the scene.
robot:
elongated
elongated
elongated
elongated unknown color
2. Tutor: "The elongated object is yellow."
robot[tutor]:
yellow
3. The asserted information is learned and verified.
shared:
visual learning
4. The visual information is updated.
yellow elongated

Fig. 17. Example of a processing pipeline. The green color represents restrictive information, while the violet color denotes assertive information. Only the beliefs and other data structures pertaining to the yellow tea box are shown.

A similar process takes place in tutor assisted learning, when the robot initiates, based on an unreliable recognition, the learning process, e.g., by asking "*G: Is this red?*." In this case, the need for assistance arises from the robot's private belief that contains the assertion about the red color and references the union representing the object. Based on this belief, the communication SA synthesizes the above question. When the robot receives the positive answer, he updates the representation of red, using a similar mechanism to that in tutor driven learning.

B. Experimental Results

The system was developed to interact with a user, and we present video evidence of the system working in this manner.[5] However, to comprehensively analyze the learning strategies, such interactive use is time consuming and impractical. We therefore, perform quantitative evaluation in simulation. The simulation environment [59] uses stored images of real scenes, which were previously captured and automatically segmented. We used a number of everyday objects, similar to those presented in Fig. 1. Each image, containing a detected and segmented object, was then manually labeled. In the learning process, the tutor is replaced by an omniscient oracle, which has the ground truth data available. In this way, extensive tests could be automated.

[5]http://cogx.eu/results

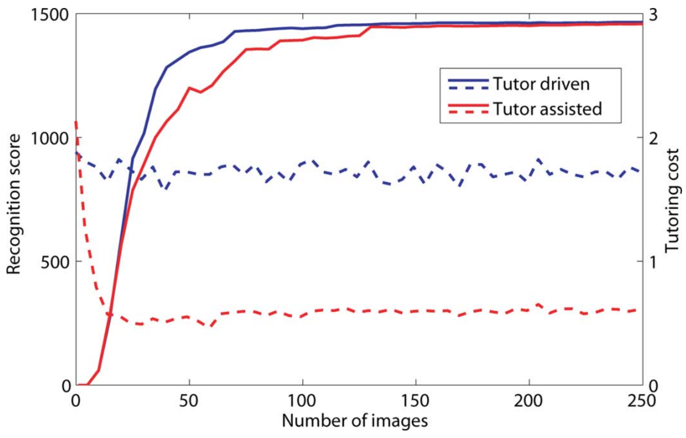


Fig. 18. Experimental results for Tutor driven and Tutor assisted learning; solid line: Recognition score, dashed line: Tutoring cost.

Six visual attributes were considered; four colors (red, green, blue, and yellow) and two shapes (elongated, compact). The database that we used for learning contains 500 images [59]. Half of the images were used to incrementally learn the representations of six visual properties, while the other 250 were used as test images. We repeated the experiment for 10 runs by randomly splitting the set of images into training and test sets and averaging the results across all runs.

During the experiment, we incrementally updated the representations with the training images for each treatment: either the Tutor driven or the Tutor assisted learning strategy. At each step, we evaluated the current knowledge by running recognition for each visual property on all test images. The learning performance was evaluated using two performance measures. The *recognition score* rewards successful recognition (true positives and true negatives) and penalizes incorrectly recognized visual properties (false positives and false negatives). The *tutoring cost* measures the level of the tutor's involvement, as defined in [59].

Fig. 18 shows the evolution of the learning performance over time for both learning strategies. The first thing to note is that the overall results improve through time. The growth of the recognition score is very rapid at the beginning when new models of newly introduced concepts are being added, and still remains positive even after all models are formed due to refinement of the corresponding representations. Both approaches reach almost optimal results at the end (recognition score 1500 in this case).

The tutor-driven approach achieves a high recognition score more quickly, whereas tutor-assisted learning generates a lower tutoring cost. One result is the flip side of the other. The tutor-driven approach ensures feedback every image, but while this speeds learning it takes time. In tutor-assisted learning, the robot only questions to begin with. Once its knowledge level has improved the number of questions drops. Tutor-assisted learning is therefore, more efficient in terms of tutoring cost, while in the long-term it produces comparable recognition results. Note that the sequence of training samples was the same in both cases; the performance of the Tutor-assisted learning could be further improved if the system could actively select learning samples, however this was not the subject of this experiment.

VIII. Related Work

We have already related each element of our approach to the literature. We now do this for the overall approach. It is important to emphasize that our focus is artificial cognitive systems. Our approach is not directly inspired by, nor intended to model, mechanisms or behaviors in humans or other animals. It may be that the core concept presented here of using representations of gaps and uncertainties in knowledge to guide the selection of learning experiences is useful to those studying biological systems, but speculation on this is beyond the scope of this paper.

There are several examples of robot systems that reason about their epistemic state. Flakey [60], used a modal logic of knowledge to create controllers with provable epistemic properties. A very different approach is the intelligent adaptive curiosity model of Oudeyer and Kaplan [61] based on the pioneering work of Schmidhuber and others [62], used rewards generated from improvements in prediction to guide exploratory motor behaviors in a robot in order to discover patterns of interaction with the environment. This approach differs significantly from ours in two ways. First, they employ what is essentially a reinforcement learning framework to control curiosity, whereas we use heuristic measures to select goals to plan for using models of epistemic effects. We are currently extending our planning framework to include a decision theoretic planner. This will be able to tradeoff the values of different exploratory actions in the plan. However, our framework plans curious behavior rather than learning it. Second, our framework uses very rich representations, and can thus be seen as one way of scaling developmental approaches to robotics.

Related to Dora there is a growing body of work on entropic methods for uncertainty reduction in robot SLAM [63]–[65]. However, our approach is the first where the robot uses logical models of its epistemic state together with planning algorithms to select learning goals. The key difference is that our systems approach means Dora can work towards many informational goals, which can all be quite different.

Related to George, there are several systems that address the problem of interactive continuous learning (e.g., [66]–[71]). Different systems focus on different aspects of the problem, such as the system architecture and integration [68], [69], [71], learning [66], [67], [71], or social interaction [70]. Our work focuses on the integration of visual and linguistic information by forming beliefs about the state of the world; these beliefs are used in the continuous learning process for updating the current representations. The system we present enables different kinds of learning in a dialog with a human teacher, including self-motivated learning, triggered by autonomous knowledge gap detection.

Self-understanding is not the only way to achieve autonomous learning. There are other systems that can create quite sophisticated representations without it. Modayil and Kuipers [72] show how a robot is able to acquire object-based representations of the world by bootstrapping representations acquired during motor babbling. They are able to obtain planning operators from interaction with a continuous world. We argue that by also employing representations of epistemic state and performing abduction on a world model the robot will explore hypotheses in a more efficient manner.

IX. Conclusion

We have presented a new way of thinking about autonomous learning that concentrates on architectures and representations. The representational component of our theory is two-fold: on the one hand we employ representations of uncertainty and gaps in different modalities; on the other we represent how that lack of knowledge may change under action. The architectural side of our theory shows how to make the different components of our approach work together. We first argued that a cognitive robot requires several working memories because we must parallelize the processing of information in different sensory modalities. We also suggested that representations of gaps in knowledge in each modality are a good way to drive curious behavior. From these two building blocks, the rest of the choices follow. Each modality specific subsystem employs its own representations of what is known, uncertain, and unknown. These different models are linked in an central multimodal model, which itself can have gaps and uncertainties. This model has symbols that are stable enough to support planning of action. Each modality specific subsystem can make suggestions to a central coordination system to pursue its preferred epistemic goals. To ensure scaling we propose that these possible learning activities must be quickly sorted, and only a subset handed to planning. We have shown how planning can be done in a way that explicitly models the epistemic effects of physical and sensing actions in an efficient way.

We have shown that this approach works, by implementing two robot systems that put these elements together. Each illustrates different aspects of our approach. Dora illustrates the architecture, representations of gaps and uncertainty in spatial representations, the goal management system, and the use of planning with epistemic goals. George illustrates how we explicitly represent uncertainty in multimodal representations of a specific situation, and uncertainty and novelty in the long term model of how different modalities are related.

What are the open research issues? First, of all our approach to novelty is limited. The work on KDE models provides a particular approach to this, in a particular domain, but it is far from complete. There is also a question about how constrained the tacit design knowledge makes the self-extension. At the moment Dora, and to a lesser extent George extend their models within knowledge spaces that are quite well defined. The Dora design tacitly assumes that placeholders will become places, and George has the visual features necessary to learn the correct associations with words describing color and shape. In addition the typology we have described for different types of incompleteness is only a beginning. Most challengingly, however, we have not yet dealt with the representation of different kinds of outcome or causal incompleteness. It is in general, very difficult to model and reason about these in worlds with noisy observations and noisy actions. This is because an unexpected outcome could be due to observation noise, action noise, or true novelty. Variations on latent variable models such as factored

POMDPs provide a probabilistic approach, but these are notoriously difficult to learn and reason with. To identify hidden causes in models of actions is also difficult. Suppose an action of a robot fails, such as a grasping action? This could be because of picking a poor grasp position, failing to grip strongly enough, or estimating wrongly where the object was. These possible causes can be distinguished if the robot has the *a priori* notion that they are possible causes of grasp failure, but in general, we want the robot to be able to discover for itself that they are possible causes. This degree of open-endedness will take many years to tackle.

In summary, we believe that our knowledge-based, architecturally coherent approach to autonomous mental development is a worthwhile one, but there are many challenges that remain.

REFERENCES

[1] N. Hawes and J. Wyatt, "Engineering intelligent information-processing systems with CAST," *Adv. Eng. Inform.*, vol. 24, pp. 27–39, 2010.

[2] N. Hawes, M. Brenner, and K. Sjöö, "Planning as an architectural control mechanism," in *Proc. 4th ACM/IEEE Int. Conf. Human Robot Inter. (HRI)*, New York, 2009, pp. 229–230.

[3] J. E. Laird, A. Newell, and P. S. Rosenbloom, "Soar: An architecture for general intelligence," *Artif. Intell.*, vol. 33, no. 3, pp. 1–64, 1987.

[4] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychol. Rev.*, vol. 111, no. 4, pp. 1036–1060, 2004.

[5] K. Talamadupula, J. Benton, P. Schermerhorn, S. Kambhampati, and M. Scheutz, "Integrating a closed world planner with an open world robot: A case study," in *Proc. AAAI Conf. Artif. Intell.*, Atlanta, GA, 2010.

[6] J. Peltason, F. H. K. Siepmann, T. P. Spexard, B. Wrede, M. Hanheide, and E. A. Topp, "Mixed-initiative in human augmented mapping," in *Proc. Int. Conf. Robotic. Autom.*, Kobe, Japan, 2009, pp. 3175–3182.

[7] H. Jacobsson, N. Hawes, G.-J. Kruijff, and J. Wyatt, "Crossmodal content binding in information-processing architectures," in *Proc. 3rd Int. Conf. Human–Robot Interact. (HRI)*, Amsterdam, The Netherlands, 2008.

[8] J. Kelleher, N. Creaney, Ed., "Integrating visual and linguistic salience for reference resolution," in *Proc. 16th Irish Conf. Artif. Intell. Cogn. Sci. (AICS-05)*, Ulster, U.K., 2005.

[9] E. Punskaya, "Bayesian Approaches to Multi-Sensor Data Fusion," M.Sc. thesis, Engineering Department, Cambridge Univ., Cambridge, U.K., 1999.

[10] R. Engel and N. Pfleger, "Modality fusion," in *SmartKom: Foundations of Multimodal Dialogue Systems*, W. Wahlster, Ed. Berlin, Germany: Springer-Verlag, 2006, pp. 223–235.

[11] D. Roy, "Semiotic schemas: A framework for grounding language in action and perception," *Artif. Intell.*, vol. 167, no. 1-2, pp. 170–205, 2005.

[12] A. Pronobis, K. Sjöö, A. Aydemir, A. N. Bishop, and P. Jensfelt, Representing Spatial Knowledge in Mobile Cognitive Systems Kungliga Tekniska Högskolan, CVAP/CAS, 2010, Tech. Rep. TRITA-CSC-CV 2010:1 CVAP 316.

[13] J. Folkesson, P. Jensfelt, and H. Christensen, "The m-space feature representation for slam," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1024–1035, Oct. 2007.

[14] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," in *Proc. 4th Int. Symp. Robot. Res.*, Santa Cruz, CA, 1987.

[15] A. Aydemir, A. Bishop, and P. Jensfelt, "Simultaneous object class and pose estimation for mobile robotic applications with minimalistic recognition," in *Proc. Int. Conf. Robot. Autom. (ICRA'09)*, Kobe, Japan, 2010.

[16] González-Banos and Laser, "A randomized art-gallery algorithm for sensor placement," in *Proc. 17th Annu. Symp. Comput. Geometry*, Medford, MA, 2001.

[17] J. Castellanos, R. Martinez-Cantin, J. Tardos, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robot. Autonom. Syst.*, vol. 55, no. 1, Jan. 2007.

[18] A. N. Bishop and P. Jensfelt, "A stochastically stable solution to the problem of robocentric mapping," in *Proc. Int. Conf. Robot. Autom. (ICRA'09)*, Kobe, Japan, 2009.

[19] A. N. Bishop and P. Jensfelt, "Stochastically convergent localization of objects and actively controllable sensor-object pose," in *Proc. 10th Eur. Contr. Conf. (ECC'09)*, Budapest, Hungary, 2009.

[20] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2009, to be published.

[21] A. Pronobis, O. M. Mozos, B. Caputo, and P. Jensfelt, "Multi-modal semantic place classification," *Int. J. Robot. Res. (IJRR)*, vol. 29, no. 2-3, pp. 298–320, Feb. 2010.

[22] H. Zender, O. M. Mozos, P. Jensfelt, G.-J. M. Kruijff, and W. Burgard, "Conceptual spatial representations for indoor mobile robots," *Robot. Autonom. Syst.*, vol. 56, no. 6, pp. 493–502, Jun. 2008.

[23] B. Kuipers, "The spatial semantic hierarchy," *Artif. Intell.*, vol. 119, no. 1-2, pp. 191–233, 2000.

[24] N. Tomatis, I. Nourbakhsh, and R. Siegwart, "Hybrid simultaneous localization and map building: A natural integration of topological and metric," *Robot. Autonom. Syst.*, vol. 44, no. 1, pp. 3–14, Jul. 2003.

[25] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, and F. Savelli, "Local metrical and global topological maps in the hybrid spatial semantic hierarchy," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA'04)*, Barcelona, Spain, 2004.

[26] J. Carlos, P. Broch, and M. E. Monferrer, "Cognitive maps for mobile robot navigation: A hybrid representation using reference systems," in *Proc. 19th Int. Workshop Qualitative Reason. (QR'05)*, Graz, Austria, May 2005.

[27] T. Spexard, S. Li, B. Wrede, J. Fritsch, G. Sagerer, O. Booij, Z. Zivkovic, B. Terwijn, and B. Kröse, "BIRON, where are you? Enabling a robot to learn new places in a real home environment by integrating spoken dialog and visual localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'06)*, Beijing, China, Oct. 2006.

[28] A. Ranganathan and F. Dellaert, "Semantic modeling of places using objects," in *Proc. Robot.: Sci. Syst. (RSS'07)*, Atlanta, GA, 2007.

[29] R. Rusu, Z. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D point cloud based object maps for household environments," *Robot. Autonom. Syst.*, vol. 56, no. 11, pp. 927–941, Nov. 2008.

[30] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernández-Madrigal, and J. González, "Multi-hierarchical semantic maps for mobile robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'05)*, Edmonton, AB, Canada, 2005.

[31] W. G. Kennedy, M. D. Bugajska, M. Marge, W. Adams, B. R. Fransen, D. Perzanowski, A. C. Schultz, and J. G. Trafton, "Spatial representation and reasoning for human-robot collaboration," in *Proc.22nd Nat. Conf. Artif. Intell. (AAAI'07)*, Vancouver, BC, Canada, 2007.

[32] S. Vasudevan, S. Gächter, V. Nguyen, and R. Siegwart, "Cognitive maps for mobile robots—An object based approach," *Robot. Autonom. Syst.*, vol. 55, no. 5, pp. 359–371, May 2007.

[33] C. Bäckström and B. Nebel, "Complexity results for SAS^+ planning," *Comput. Intell.*, vol. 11, no. 4, pp. 625–655, 1995.

[34] M. Brenner and B. Nebel, "Continual planning and acting in dynamic multiagent environments," *J. Autonom. Agents Multiagent Syst.*, vol. 19, no. 3, pp. 297–331, 2009.

[35] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi, *Reasoning About Knowledge*. Cambridge, MA: MIT Press, 1995.

[36] H. J. Levesque, "What is planning in the presence of sensing?," in *Proc. AAAI Conf. Artif. Intell. (AAAI-96)*, Portland, OR, 1996, pp. 1139–1146.

[37] R. Petrick and F. Bacchus, "A knowledge-based approach to planning with incomplete information and sensing," in *Proc. 6th Int. Conf. Artif. Intell. Plan. Syst. (AIPS-02)*, Toulouse, France, 2002.

[38] B. J. Grosz and S. Kraus, "Collaborative plans for complex group action," *Artif. Intell. J.*, vol. 86, 1996.

[39] K. E. Lochbaum, "A collaborative planning model of intentional structure," *Comput. Ling.*, 1998.

[40] M. Kristan and A. Leonardis, "Multivariate online kernel density estimation," in *Proc. Comput. Vis. Winter Workshop*, Nove Hrady, Czech Republic, 2010, pp. 77–86.

[41] M. Kristan, D. Skočaj, and A. Leonardis, "Online kernel density estimation for interactive learning," *Image Vis. Comput.*, pp. 1106–1116, 2009.

[42] D. Skočaj, M. Kristan, and A. Leonardis, "Continuous learning of simple visual concepts using Incremental Kernel Density Estimation," in *Proc. 3rd Int. Conf. Comput. Vis. Theory Appl. (VISSAP)* , Madeira, Portugal, 2008, pp. 598–604.

[43] X. Zhu, P. Zhang, X. Lin, and Y. Shi, "Active learning from stream data using optimal weight classifier ensemble," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, no. 99, pp. 1–15, 2010.

[44] W. Fan, Y. Huang, H. Wang, and P. S. Yu, "Active mining of data streams," in *Proc. 4th SIAM Int. Conf. Data Mining*, Orlando, FL, 2004, pp. 457–461.

[45] S. Huang and Y. Dong, "An active learning system for mining time-changing data streams," *Intell. Data Anal.*, vol. 11, no. 4, pp. 401–419, 2007.

[46] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, "Active learning with gaussian processes for object categorization," in *Proc. Int. Conf. Comput. Vis.*, Rio de Janeiro, Brazil, 2007.

[47] L. P. Beaudoin and A. Sloman, A. Sloman, D. Hogg, G. Humphreys, A. Ramsay, and D. Partridge, Eds., "A study of motive processing and attention," in *Proc. Prospects Artif. Intell. (AISB-93)*, Amsterdam, The Netherlands, 1993, pp. 229–238.

[48] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic motivation systems for autonomous mental development," *IEEE Trans. Evol. Comput.*, vol. 11, pp. 265–286, Nov. 2007.

[49] M. Wooldrige*, An Introduction to MultiAgent Systems*, 2nd ed. New York: Wiley, 2009.

[50] A. S. Rao and M. P. Georgeff, "Bdi-agents: From theory to practice," in *Proc. 1st Int. Conf. Multiagent Syst.*, San Francisco, CA, 1995.

[51] A. Patalano and C. Seifert, "Opportunistic planning: Being reminded of pending goals," *Cogn. Psychol.*, vol. 34, pp. 1–36, 1997.

[52] A. M. Coddington, "Integrating motivations with planning," in *Proc. 6th Int. Joint Conf. Autonom. Agent. Multi-Agent Syst. (AAMAS '07)*, Honolulu, HI, 2007, pp. 850–852.

[53] R. S. N. Menkes van den Briel and S. Kambhampati, "Over-subscription in planning: A partial satisfaction problem," in *Proc. ICAPS 2004 Workshop Integr. Plan. Scheduling*, Whistler, BC, Canada, 2004.

[54] P. W. Schermerhorn and M. Scheutz, "The utility of affect in the selection of actions and goals under real-world constraints," in *Proc. 2009 Int. Conf. Artif. Intell. (ICAI '09)*, Las Vegas, NV, 2009, pp. 948–853.

[55] M. Ozuysal, P. Fua, and V. Lepetit, "Fast keypoint recognition in ten lines of code," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Minneapolis, MN, 2007, pp. 1–8.

[56] M. Hanheide, N. Hawes, J. Wyatt, M. Göbelbecker, M. Brenner, K. Sjöö, A. Aydemir, P. Jensfelt, H. Zender, and G.-J. Kruijff, "A framework for goal generation and management," in *Proc. AAAI Workshop Goal-Directed Autonomy*, 2010.

[57] D. Skočaj, M. Janiček, M. Kristan, G.-J. M. Kruijff, A. Leonardis, P. Lison, A. Vrečko, and M. Zillich, "A basic cognitive system for interactive continuous learning of visual concepts," in *Proc. ICRA 2010 Workshop Interact. Commun. Autonom. Intell. Robot. (ICAIR).*, Anchorage, AK, May 2010.

[58] A. Vrečko, D. Skočaj, N. Hawes, and A. Leonardis, "A computer vision integration model for a multi-modal cognitive system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, St. Louis, MO, Oct. 2009, pp. 3140–3147.

[59] D. Skočaj, M. Kristan, and A. Leonardis, "Formalization of different learning strategies in a continuous learning framework," in *EPIROB'09*, Venice, Italy, 2009, pp. 153–160.

[60] S. Rosenschein and L. Kaelbling, "The synthesis of digital machines with provable epistemic properties," in *Proc. 1986 Conf. Theoretical Aspects Reason. About Knowledge*, 1986, pp. 83–98.

[61] P. Oudeyer, F. Kaplan, and V. Hafner, "Intrinsic motivation systems for autonomous mental development," *IEEE Trans. Evol. Comput.*, vol. 11, pp. 265–286, Nov. 2007.

[62] J. Schmidhuber, "Curious model building control systems," in *Proc. Int. Joint. Conf. Neural Netw.*, 1991, vol. 2, pp. 1458–1463.

[63] S. Thrun, W. Burgard, and D. Fox*, Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. Cambridge, MA: MIT Press, 2005.

[64] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using Rao-Blackwellized particle filters," in *Robotics Science and Systems*. Cambridge, MA: , 2005, pp. 65–72.

[65] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst (IROS)*, Lausanne, Switzerland, 2002, pp. 540–545.

[66] D. K. Roy and A. P. Pentland, "Learning words from sights and sounds: A computational model," *Cogn. Sci.*, vol. 26, no. 1, pp. 113–146, 2002.

[67] L. Steels and F. Kaplan, "AIBO's first words, the social learning of language and meaning," *Evol. Commun.*, vol. 4, no. 1, pp. 3–32, 2000.

[68] C. Bauckhage, G. Fink, J. Fritsch, F. Kummmert, F. Lomker, G. Sagerer, and S. Wachsmuth, "An integrated system for cooperative man-machine interaction," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom.*, Seoul, Korea, 2001, pp. 320–325.

[69] B. Bolder, H. Brandl, M. Heracles, H. Janssen, I. Mikhailova, J. Schmudderich, and C. Goerick, "Expectation-driven autonomous learning and interaction system," in *Proc. 8th IEEE-RAS Int. Conf. Humanoid Robot.*, Dec. 2008, pp. 553–560.

[70] A. L. Thomaz and C. Breazeal, "Experiments in socially guided exploration: Lessons learned in building robots that learn with and without human teachers," *Connect. Sci.*, vol. 20, no. 2 3, pp. 91–110, Jun. 2008.

[71] S. Kirstein, A. Denecke, S. Hasler, H. Wersing, H.-M. Gross, and E. Körner, "A vision architecture for unconstrained and incremental learning of multiple categories," *Memetic Comput.*, vol. 1, pp. 291–304, 2009.

[72] J. Modayil and B. Kuipers, "The initial development of object knowledge by a learning robot," *Robot. Autonom. Syst.*, vol. 56, no. 11, pp. 879–890, 2008.

Jeremy L. Wyatt received the Ph.D. degree in artificial intelligence from the University of Edinburgh, Edinburgh, U.K.

Since 1997, he has worked as a Senior Lecturer at the University of Birmingham, Birmingham, U.K., where he leads the FP7 project CogX. His research interests include machine learning and cognitive robotics.

Alper Aydemir is working towards the Ph.D. degree at the Royal Institute of Technology, Stockholm, Sweden, where he is working on active visual search.

His research interests include active visual search, 3-D semantic mapping, and the development of spatial representations for cognitive mobile robots.

Michael Brenner received the M.Sc. degree in cognitive science from LIMSI, Paris, France in 1999, and is currently working towards the Ph.D. degree at the Albert-Ludwigs Universität, Freiburg, Germany, working on automated planning.

He is currently a University Assistant in the Artificial Intelligence Research Group headed by Prof. Bernhard Nebel.

Marc Hanheide received the Dipl. and Ph.D. degrees in computer science from Bielefeld University, Bielefeld, Germany, in 2001 and 2006, respectively.

He is currently a Research Fellow in the School of Computer Science at the University of Birmingham, Birmingham, U.K., working in the EU integrated project CogX. In 2001, he joined the Applied Informatics Group at Bielefeld University where he worked in the EU projects VAMPIRE and COGNIRON. He is also a PI in a number of projects funded by the CoRLab and the Cluster of Excellence CITEC, Bielefeld. His research interests include autonomous robots, human–robot interaction, and architectures for cognitive systems.

Nick Hawes received the Ph.D. degree from the University of Birmingham, Birmingham, U.K., in 2004, studying the integration of anytime planning into agent architectures and has since extended this work to look at the general problems of integrating symbolic decision making into architectures for embodied intelligent systems.

He is currently a Lecturer in the Intelligent Robotics Lab, School of Computer Science at the University of Birmingham. His research interests include software engineering for cognitive systems and models of memory and attention.

Patric Jensfelt received the M.Sc. and Ph.D. degrees from Kungliga Tekniska Högskolan (KTH), Stockholm, Sweden, in 1996 and 2001, respectively.

After completing the Ph.D. degree, he worked on two industrial projects before returning to research. He is currently an Assistant Professor at the Centre for Autonomous Systems, KTH. His research interests include spatial modeling, navigation, and systems integration.

Matej Kristan received the Ph.D. degree in electrical engineering from the University of Ljubljana, Ljubljana, Slovenia, in 2008.

He is currently a Researcher at the Visual Cognitive Systems Laboratory at the Faculty of Computer and Information Science, and an Assistant Professor at the Faculty of Electrical Engineering, University of Ljubljana. He has received several awards for his research in the field of computer vision and pattern recognition. His research interests include probabilistic methods for computer vision and pattern recognition with focus on tracking, probabilistic dynamic models, and statistical online learning.

Geert-Jan M. Kruijff received the M.Sc. degree in philosophy and computer science from the University of Twente, Twente, The Netherlands, and the Ph.D. degree in mathematical linguistics from Charles University, Prague, Czech Republic.

He is currently a Senior Researcher at the DFKI Language Technology Lab, where he leads efforts in the area of "talking robots." From 2001 until 2004, he held a project leader position at Saarland University, Saarbrücken, Germany. He is particularly interested in developing theories and has implemented architectures for cognitive robots to understand and produce situated dialogue with human users. He has over 90 refereed conference papers and articles in human–robot interaction, and formal and computational linguistics.

Pierre Lison received the M.Sc. degree in computer science and engineering from the University of Louvain, Belgium, and the M.Sc. degree in computational linguistics from the University of Saarland, Saarbrücken, Germany. He is currently working towards the Ph.D. degree in adaptive dialog management, under the direction of Geert-Jan M. Kruijff.

He is currently a Researcher at the German Research Centre for Artificial Intelligence, Saarbrücken, Germany. He is also involved in several international projects in cognitive robotics and human–robot interaction.

Andrzej Pronobis is currently working towards the Ph.D. degree at the Royal Institute of Technology (KTH), Stockholm, Sweden where he is working on spatial understanding with particular focus on multimodal place classification for topological localization and semantic knowledge extraction in robotic systems.

Kristoffer Sjöö received the M.Sc. degree in applied physics and electrical engineering from Linköping University, Linköping, Sweden. He is currently working towards the Ph.D. degree at the Royal Institute of Technology, Stockholm, Sweden, working on spatial knowledge representation for mobile robots.

Alen Vrečko is working towards the Ph.D. degree at the Visual Cognitive Systems Laboratory, University of Ljubljana, Ljubljana, Slovenia.

He is currently a Researcher at the University of Ljubljana. His research interests include cognitive systems, cognitive computer vision, and cross-modal learning.

Hendrik Zender received the Dipl. degree in computational linguistics from Saarland University, Saarbrücken, Germany, in 2006. He is currently a Ph.D. candidate and Researcher at DFKI, Saarbrücken, Germany, working on conceptual mapping and situated dialogue in human–robot interaction.

Michael Zillich received the Ph.D. degree from the Vienna University of Technology, Vienna, Austria, in 2007.

He spent a year as a Research Fellow at the University of Birmingham, Birmingham, U.K., before returning as a Research Fellow to Vienna. His research interests include vision for robotics and cognitive vision.

Danijel Skočaj received the Ph.D. degree from the University of Ljubljana, Ljubljana, Slovenia, in 2003.

He is currently an Assistant Professor at the Faculty of Computer and Information Science, University of Ljubljana. His research interests lie in the fields of cognitive vision and cognitive systems and include automatic modeling of objects and scenes with the emphasis on continuous and interactive visual learning and recognition.