Mixed Logical Inference and Probabilistic Planning for Robots in Unreliable Worlds

Shiqi Zhang*, Mohan Sridharan†, Jeremy Wyatt‡

*Department of Computer Science, The University of Texas at Austin, USA.
szhang@cs.utexas.edu
†Department of Electrical and Computer Engineering, The University of Auckland, NZ.
m.sridharan@auckland.ac.nz
‡School of Computer Science, The University of Birmingham, UK.
jlw@cs.bham.ac.uk

***Abstract*—Deployment of robots in practical domains poses key knowledge representation and reasoning challenges. Robots need to represent and reason with incomplete domain knowledge, acquiring and using sensor inputs based on need and availability. This paper presents an architecture that exploits the complementary strengths of declarative programming and probabilistic graphical models as a step towards addressing these challenges. Answer Set Prolog (ASP), a declarative language, is used to represent, and perform inference with, incomplete domain knowledge, including default information that holds in all but a few exceptional situations. A hierarchy of partially observable Markov decision processes (POMDPs) probabilistically models the uncertainty in sensor input processing and navigation. Non-monotonic logical inference in ASP is used to generate a multinomial prior for probabilistic state estimation with the hierarchy of POMDPs. It is also used with historical data to construct a Beta (meta) density model of priors for metareasoning and early termination of trials when appropriate. Robots equipped with this architecture automatically tailor sensor input processing and navigation to tasks at hand, revising existing knowledge using information extracted from sensor inputs. The architecture is empirically evaluated in simulation and on a mobile robot visually localizing objects in indoor domains.**

I. Introduction

Mobile robots are increasingly being deployed in practical application domains such as healthcare, disaster rescue and navigation. These robots receive far more raw data from sensors than is possible to process in real-time, and it is difficult to equip the robots with accurate and complete domain knowledge. Human participants, if any, may not have the time and expertise to provide elaborate and accurate feedback. Furthermore, the descriptions of knowledge and uncertainty obtained from different sources may complement or contradict each other. Widespread deployment of robots thus poses the fundamental challenge of enabling them to represent and reason with qualitative and quantitative descriptions of incomplete domain knowledge and the associated uncertainty, acquiring and using sensor inputs based on need and availability.

Although probabilistic graphical models such as partially observable Markov decision processes (POMDPs) have been used to plan sensing and navigation on robots by probabilistically modeling the associated uncertainty, it is difficult to represent and reason with commonsense knowledge in such formulations. Declarative languages such as Answer Set Prolog (ASP) are well-suited for knowledge representation and non-monotonic logical reasoning, but they do not support probabilistic modeling of uncertainty [9]. Prior work integrating ASP with hierarchical POMDPs [32] did not support key capabilities such as default reasoning, incremental bidirectional flow of information between the commonsense inference and probabilistic reasoning components, and metareasoning with observations and historical data. The architecture described in this paper addresses these limitations by making the following novel contributions:

- Richer representation and inference in ASP with incomplete domain knowledge, which includes default information that holds in all but a few exceptional situations, to effectively reduce the task completion time.
- Use of ASP-based inference to heuristically generate a multinomial prior for the POMDP state estimation that is used to plan sensing and navigation, with the subsequent observations adding relevant statements to the ASP knowledge base.
- Metareasoning with observations and a Beta density model of priors based on historical data, supporting early termination of tasks that cannot be accomplished with the existing models.

The architecture thus establishes a continuous loop of non-monotonic logical inference, probabilistic planning and incremental knowledge revision. The architecture is grounded and evaluated in simulation and on mobile robots localizing (i.e., determining the location of) objects in indoor domains.

II. Related Work

Researchers have used probabilistic graphical models such as POMDPs to formulate planning, sensing, navigation and interaction on robots [11], [16], [24]. However, these formulations, by themselves, are not well-suited for commonsense reasoning. In parallel, research in classical planning has provided sophisticated algorithms for knowledge representation (KR) and logical reasoning [10], which have been used on mobile robots [8]. However, these algorithms typically require a significant amount of prior knowledge regarding the domain, and the preconditions and effects of the actions. Many

algorithms also do not support merging of new, unreliable information (e.g., from sensors) with the current beliefs in a knowledge base. Answer Set Prolog (ASP), a non-monotonic logic programming paradigm, is well-suited for representing and reasoning with commonsense knowledge [2], [9]. It has been used in cognitive robotics [7], e.g., for reasoning by simulated robot housekeepers [6] and for representing domain knowledge learned through natural language processing [4]. However, ASP does not support quantitative models of uncertainty, whereas a lot of information available to robots is represented probabilistically so as to quantitatively model the uncertainty in sensing and acting.

Robotics researchers have developed algorithms that support logical and probabilistic reasoning for task, motion, or behavior planning [11], [17]. Semantic maps and commonsense knowledge have been used with probabilistic algorithms to locate targets, and for open world planning [14], [15]. Declarative programming and continuous-time planners have been used for path planning in mobile robot teams [29], and a probabilistic extension of ASP has been combined with POMDPs for commonsense inference and probabilistic planning in human-robot dialog [35]. Principled algorithms developed to combine logical and probabilistic reasoning include the Markov logic network that combines probabilistic graphical models and first order logic, assigning weights to logic formulas [23]; and Bayesian Logic that relaxes the unique name constraint of first-order probabilistic languages to provide a compact representation of distributions over varying sets of objects [20]. Other examples include independent choice logic [21], PRISM [12], probabilistic first-order logic [13], first-order relational POMDPs [25], and probabilistic extensions to ASP [3], [18]. However, these algorithms are limited in their ability to support the desired knowledge representation and reasoning capabilities for human-robot collaboration. Algorithms based on first-order logic do not provide the desired expressiveness for capabilities such as default reasoning, e.g., it is not always possible to express degrees of belief quantitatively. Other algorithms based on logic programming do not support one or more of the capabilities such as: reasoning about relations as in causal Bayesian networks; incremental addition of probabilistic information; reasoning with large probabilistic components; or dynamic addition of variables with different ranges [3]. The architecture described in this paper is a step towards achieving these capabilities. Key limitations of prior work on integrating ASP and POMDPs [32] are addressed by supporting default reasoning, generating priors based on ASP inference for POMDP state estimation, and metareasoning with observations and historical data from comparable domains. Preliminary versions of some of these contributions are documented in workshop papers [30], [31]. This paper provides a detailed description of the novel contributions, supported by extensive experimental evaluation in simulation and on a mobile robot.

III. Problem Formulation

Figure 1 depicts the control architecture, whose components are illustrated and evaluated in this paper for *visual target localization*. A mobile robot determines the locations of desired objects in an indoor domain using (primarily) visual data. It is assumed that the robot revises the domain map and estimates its own location using laser range data, and has learned object models and semantic labels for rooms.

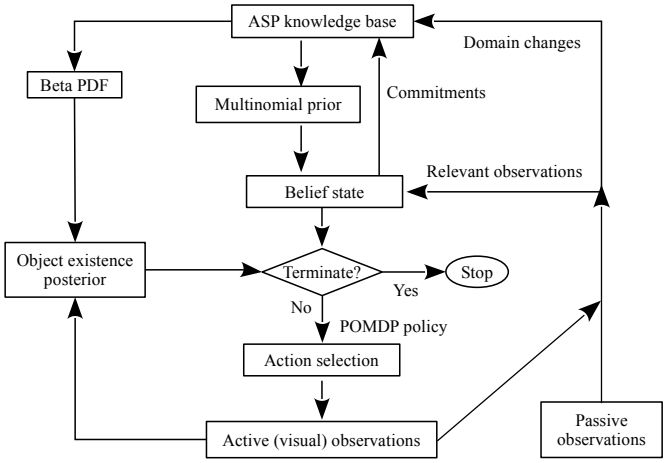


Fig. 1: Architecture integrates knowledge representation, non-monotonic logical inference and probabilistic planning.

The ASP *Knowledge Base* (KB) contains statements describing domain objects and relations between them, including default information that holds in all but a few exceptional situations. Currently, some statements are hand-coded (e.g., axioms), while others are learned from sensor inputs and historical data. For any given task, inference in the KB provides an *Answer Set*, a set of ground literals representing the current beliefs based on non-monotonic logical inference in the KB (Section III-A). In parallel, the given task (e.g., to localize a specific object) is formulated as a POMDP that probabilistically captures the uncertainty in sensing and navigation (Section III-B). The answer set heuristically generates a multinomial prior for the POMDP state estimation, and action selection is based on the posterior distribution (Section III-C). The answer set and historical data from comparable domains also populate a Beta density that defines a prior for metareasoning with observations in the current domain, supporting early termination of tasks when appropriate (Section III-D). A robot equipped with this architecture obtains observations from algorithms activated when needed (e.g., for visual object recognition) and algorithms that are always in use (e.g., obstacle avoidance using range data). Relevant observations (e.g., of the target object) update the POMDP belief distribution, and a belief with high certainty commits an appropriate statement to the ASP KB. Some observations may also identify domain changes, e.g., using range data to identify changes in the map of the domain, which are also used to revise the KB. If the revised KB provides a new multinomial prior, it is combined with the likelihood of the observation sequence to obtain the revised posterior for action selection. The following sections focus on the new contributions of this paper; other components are summarized for completeness. For target localization, inference in the ASP KB is at the coarser resolution of rooms or places, while the POMDP solver works at the finer resolution of grid cells in rooms.

A. Knowledge Representation with ASP

Answer Set Prolog (ASP) is a declarative language that can represent recursive definitions, defaults, causal relations, special forms of self-reference, and language constructs that occur frequently in non-mathematical domains, and are difficult to express in classical logic formalisms [2]. ASP is based on the stable model (answer set) semantics of logic programs and research in non-monotonic logics [9]. ASP can draw conclusions due to lack of evidence to the contrary, using concepts such as *default negation* (negation by failure) and *epistemic disjunction*. For instance, unlike "$\neg$a", which implies that "*a is believed to be false*", "`not a`" only implies that "*a is not believed to be true*"; and unlike "`p` $\vee\neg$`p`" in propositional logic, "`p or` $\neg$`p`" is not a tautology. ASP also supports non-monotonic reasoning—adding a statement can reduce the set of inferred consequences—reasoning in large KBs, and reasoning with quantifiers. These capabilities have led to the use of ASP by an international research community.

The following basic definitions will be used in this paper [9]. Variable and object constants are *terms*, and a function of terms is a term; terms with no symbols and no variables are *ground*. A predicate of terms is an *atom*; it is *ground* if all its terms are ground. An atom or its negation is a *literal*: ground atoms and their negations are *ground literals*. *Statics* are domain properties whose truth values cannot be changed by actions, and *fluents* are properties that can be changed by actions. A *basic fluent*, also called an inertial fluent in the knowledge representation literature, is subject to inertial laws and can be directly changed by actions, while a *defined fluent* cannot be directly changed by an action, is defined in terms of other fluents, and is not subject to laws of inertia.

An ASP *program* (Π) has a *sorted signature* Σ and axioms of the form: l_0 or $\cdots$ or $l_i \leftarrow l_{i+1},\cdots,l_m$, not $l_{m+1},\cdots,$ not l_n. Each l in the axiom is a literal of Σ. The *sorts* in the illustrative example are: *object*, *class*, and *room*; sorts can have subsorts, e.g., *fridge*, *printer*, and *book* are subsorts of *object*. $\Sigma = \langle \mathscr{O}, \mathscr{F}, \mathscr{P}, \mathscr{V} \rangle$ defines the names of objects[1], functions, predicates and variables available for use. Each function or predicate is defined in terms of the sorts of its arguments, e.g., predicate `in(object, room)` can represent the relation `in(fridge1, kitchen)`. Program Π is thus a collection of statements describing domain objects and relations between them. The ground literals in an answer set obtained by solving Π represent beliefs of an agent associated with Π. Since program consequences are statements that are true in all such belief sets, the following discussion assumes that inference in the ASP KB produces only one answer set.

Unlike prior work that combined ASP and POMDPs [32], the KB in this paper includes default knowledge and relationships in a complex domain, e.g., the simulated domain in Figure 2, and the effects of incremental knowledge revision are analyzed thoroughly. The KB includes a hierarchy of object classes; leaf nodes are object instances, and parents of leaf nodes are *primary classes*. Information extracted from historical data helps identify some relations between object classes, creating some nodes and links between the root node and primary classes. Robots use information extracted from sensor inputs to add object instances and revise the KB.

[1]Unlike the sort *object*, elements of $\mathscr{O}$ are object constants (or symbols).

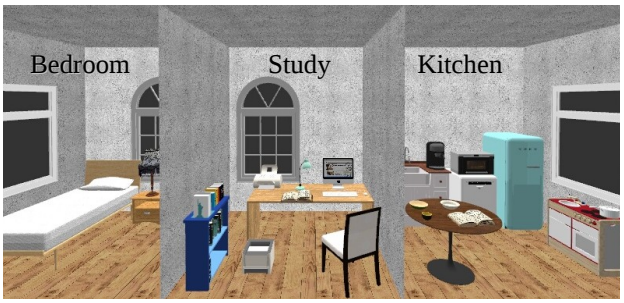


Fig. 2: Illustrative simulated domain used for experimental evaluation, with a bedroom, study and kitchen. The computer, fax machine, and printer are usually in the study; books are on the bookshelf; and kitchenware is in the kitchen. However, there are some exceptions, e.g., cookbooks are in the kitchen.

Predicates in the KB are applied recursively when appropriate. The statics of the domain include: `is(object, class)`, which describes class membership of an object, and `subclass(class, class)`, which describes class hierarchy. The basic fluents of the domain include: `in(object, room)`, which describes the room location of an object, `accessible(room)`, which states if a room is accessible, and `on(object, object)`, which states if an object is on another object. The defined fluent `exists(class, room)` implies that an instance of a specific class exists in a specific room. The sort *step* is included for temporal reasoning and the relation `holds(fluent, step)` implies that a particular fluent holds true at a particular timestep. The KB includes *reasoning* rules such as:

(1) $\texttt{holds(exists(C,R),I)} \leftarrow \texttt{holds(in(O,R),I)}, \texttt{is(O,C)}.$

(2) $\texttt{holds(exists(C1,R),I)} \leftarrow \texttt{holds(exists(C2,R),I)}, \texttt{subclass(C2,C1)}.$

(3) $\neg\texttt{holds(in(O,R2),I)} \leftarrow \texttt{holds(in(O,R1),I)}, \ \texttt{R1!=R2}.$

The first rule states that if an object O of class C is in room R, an object of class C is inferred to exist in R; the second rule applies the existence predicate recursively in the class hierarchy; and the third rule states that an object's location is unique. The KB also includes the closed world assumption for defined fluents, and *inertial axioms* that state that the value of a basic fluent `F` remains unchanged unless there is explicit evidence to the contrary:

$$\texttt{holds(F,I+1)} \leftarrow \texttt{holds(F,I)}, \texttt{ not } \neg\texttt{holds(F,I+1)}.$$
$$\neg\texttt{holds(F,I+1)} \leftarrow \neg\texttt{holds(F,I)}, \texttt{ not holds(F,I+1)}.$$

As an example of non-monotonic reasoning in ASP, consider an ASP program that includes statements: `step(1..2)`, `is(prml, book)`[2], and `holds(in(prml, study), 1)`. Inference produces the answer set[3] with statements (excluding existing statements): `holds(in(prml, study), 2)` and `holds(exists(book, study), 2)`. However, adding the statement: `holds(in(prml, bedroom), 2)` results in an

[2]The "prml" is a specific book: *Pattern Recognition and Machine Learning*.

[3]We use SPARC [1] to solve ASP programs, as described later.

answer set that revises the outcomes of the previous inference step by adding: ¬ holds(in(prml, study), 2) and holds(exists(book, bedroom), 2).

Robots collaborating with humans frequently receive domain knowledge that is true in all but a few exceptional situations. An example of such *default* domain knowledge in the simulated domain of Figure 2 would be "*the microwave is usually in the kitchen*". Although such (qualitative) commonsense knowledge can be very useful, meaningful representation of, and reasoning with, such knowledge is challenging. For instance, if the logical statement corresponding to a default is assigned a high probability, the robot's performance (based on this knowledge) may be sensitive to the choice of this probability, and it will be difficult to represent exceptions to such defaults. ASP provides an elegant representation for defaults and exceptions (if any). One significant addition to the ASP component of the architecture is the inclusion of such default knowledge about object locations. Consider the statement: "*books are typically in the study*" which can be represented in ASP as follows:

$$\texttt{in(X,study)} \leftarrow \texttt{book(X)},\quad \texttt{not ab}(\texttt{d}_{in}\texttt{(X)}),\\ \texttt{not } \neg\texttt{in(X,study)}.$$

where ab(d(X)) implies "X is abnormal with respect to d" and supports the encoding of exceptions to defaults. For instance, while textbooks are likely to be in the study, cookbooks are more likely to be in the kitchen. We can first encode the class hierarchy of books in the KB:

$$\texttt{book(X)} \leftarrow \texttt{textbook(X)}.\\ \texttt{book(X)} \leftarrow \texttt{cookbook(X)}.$$

We can then encode *weak exceptions* and a *strong exception* to the default as follows:

$\texttt{ab}(\texttt{d}_{in}\texttt{(X)}) \leftarrow \texttt{cookbook(X)}.$ %weak exception
$\texttt{ab}(\texttt{d}_{in}\texttt{(X)}) \leftarrow \texttt{not } \neg\texttt{cookbook(X)}, \texttt{book(X)}.$ %weak exception
$\neg\texttt{in(X,study)} \leftarrow \texttt{cookbook(X)}.$ %strong exception

where, the two forms of the weak exception render the default inapplicable, while the strong exception directly falsifies the default. Assume that the weak exception has been included in the KB and consider the following statements:

$$\texttt{textbook(prml)}.\\ \texttt{cookbook(spices)}.$$

Inference produces: in(prml, study) *but does not make any claim about* the location of spices, i.e., it is *unknown* if this cookbook is in the study or not. For visual target localization, the KB includes information about the default locations of objects—see Section IV.

Inconsistencies caused by the addition of incorrect information to the ASP KB can be corrected by subsequent sensor inputs. ASP also provides planning and diagnosis capabilities [9] not used in this paper but included in other work [33]. Although ASP has been used in the development of agent architectures, ASP does not support probabilistic modeling of uncertainty, and architectures that combine ASP with probabilistic reasoning lack key representation and reasoning capabilities (see Section II). The contributions of this paper are a significant step towards addressing these limitations.

B. Planning under Uncertainty with POMDPs

A robot that can localize itself has to account for the uncertainty in navigation and sensing as it moves and analyzes images of specific scenes to accurately localize an object. The robot must also pick a sequence of places to search; within the Bayesian framework, the active sensing, information processing and navigation are formulated as a probabilistic sequential decision making task, and more specifically as a POMDP. Since it is computationally intractable to solve (and plan with) practical-sized POMDPs in real-time, our prior work introduced a hierarchical decomposition of the POMDP formulation [34]—Figure 3 summarizes this decomposition. For a specific target, the 3D area is represented as a discrete 2D grid, each grid cell storing the probability of target existence. The visual search (VS)-POMDP plans an action sequence to analyze a sequence of scenes, with the objective of maximizing the information gain. For each scene, the scene processing (SP)-POMDP plans the processing of regions of images of the scene using available algorithms. This hierarchical decomposition supports automatic belief propagation between the levels of the hierarchy and automatic model creation at each level [26], [34]. Thus, ASP-based inference operates at the (abstract) level of rooms, and POMDPs plan at the higher resolution of cells. The salient features of the hierarchy of POMDPs are described briefly for completeness.

For locating a specific object in a grid with N cells, the VS-POMDP is the tuple $\langle S,A,Z,T,O,R\rangle$. Each entry in the set of states S corresponds to the event that the target is in a specific grid cell, and executing one of the actions in A causes the robot to move and analyze a specific cell[4]; $Z:\{\text{present, absent}\}$ is the observation set that indicates if the target is detected. $T: S\times A\times S'\rightarrow[0,1]$ is the state transition function, and $O: S\times A\times Z\rightarrow[0,1]$ is the observation function. Since the state is not directly observable, the robot maintains a probability distribution b over the states; each entry $b_i, i\in[1,N]$ of this *belief state* is the probability of the corresponding state s_i. Uncertainty in the belief distribution is measured by computing its entropy. To maximize information gain, the reward for action a_t is defined as the actual reduction in entropy between belief state b_t and the resultant belief state b_{t+1}. Thus $R: B\times B'\rightarrow\mathbb{R}$ is the reward specification, where B is the space of belief states. The observation function is learned by the robot as a function of its position, the target's position, the camera's field of view, and the observation functions of the hierarchy's lower levels. Given the tuple, a POMDP solver can be used to compute a *policy* that maps belief states to actions by minimizing entropy over a planning horizon. This formulation can become computationally intractable for real-time operation because the number of grid cells can increase significantly in complex domains. Our previous work [34] addressed this challenge by enabling robots to learn a *convolutional policy kernel* from the policy for a small region, exploiting the rotation and shift invariance properties of visual search. This kernel is convolved with larger maps to efficiently generate appropriate policies. Furthermore, movement between grid cells is assigned a cost

[4]The set A also includes terminal actions to terminate plan execution

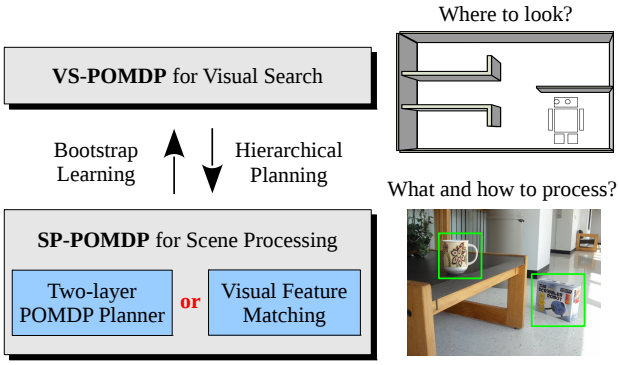


Fig. 3: Overview of the hierarchy of POMDPs for acquiring and processing visual inputs for target localization.

proportional to the distance to be traveled.

For any chosen scene, the SP-POMDP plans the sequence of visual input processing algorithms to be executed on a sequence of salient regions of interest (ROIs) in images of the scene. The SP-POMDP may have one or two layers depending on scene complexity, i.e., the number of ROIs and types of features extracted from images of the scene. For instance, each ROI extracted from an image of the scene is modeled as a lower-level (LL) POMDP. Each LL policy provides the sequence of algorithms to apply on a specific ROI to detect the desired object, e.g., algorithms to determine the dominant color or shape in the ROI. LL policies of all image ROIs are used to automatically create a high-level (HL) POMDP. Executing an action in the HL policy directs attention to a specific ROI. Executing the corresponding LL policy (until termination) provides an observation that causes an HL belief update and an action choice. These steps are repeated until a decision is made about the presence or absence of the target in the image. This decision provides an observation in the VS-POMDP, resulting in a belief update and an action choice in the form of a scene for subsequent analysis. This process continues until the belief of the target's presence in a grid cell exceeds a preset threshold (i.e., robot claims that the target has been found and localized), or a time limit is exceeded (i.e., target is not found). The entire hierarchy is tailored automatically to tasks at hand—see [26], [34] for details.

C. Integrating Logical and Probabilistic Beliefs

The answer set obtained by inference in the ASP KB represents the current, logically-expressed beliefs of the robot (Section III-A), which can be used to guide the probabilistic planning of sensor input processing and navigation. However, these beliefs are not compatible with the probabilistic belief distributions used by the hierarchy of POMDPs (Section III-B). Previous work heuristically generated an ASP-based belief distribution from a predominantly static KB, and used a generalized form of linear and logarithmic averaging methods (r-norm) [5] for weighted averaging of this belief distribution and the belief distribution modeled by the POMDPs [32]. In this paper, we present an approach that supports an incremental, bidirectional flow of information between the commonsense inference and probabilistic reasoning components—the approach consists of two steps: (1) the count of relevant literals in the answer set is used to (heuristically) create a multinomial prior over rooms the target may be in (Section III-C1); and (2) the prior and an incrementally populated observation likelihood (at the level of cells) are used for POMDP state estimation, resulting in a posterior belief distribution that is used for subsequent action selection (Section III-C2).

1) **Generating a Multinomial Prior from an Answer Set:** The conversion of relevant literals in an answer set to a multinomial (probabilistic) prior over rooms is based on: (a) knowledge of object classes and of specific object instances in the domain; and (b) postulates that capture object co-occurrence relationships. This paper illustrates this approach for visual target localization—some postulates (and their representation) may need to be revised for other sensors or domains.

Postulate 1: *Existence of objects of a primary class (in a room) provides support for the existence of other objects of this class (in the room).* The level of support is proportional to the logarithm of the number of objects, inspired by **Fechner's law**[5], which states that subjective sensation is proportional to the logarithm of stimulus intensity:

$$perception = ln(stimulus) + const \tag{1}$$

This law has been applied to visual processing [28] and explored in our previous work; here, we adapt it for the primary source of information (visual cues). The support for the existence of a specific target object in a room is given by:

$$\psi_n = \begin{cases} 0 & if \quad a_n = 0 \\ ln(a_n) + \xi & otherwise \end{cases} \tag{2}$$

where a_n is the number of (known) objects of the primary class (of the target object) in the room, and $\xi = 1$ corresponds to *const* in Equation 1. If there is only one instance of certain objects in the domain (e.g., a fridge), this can be modeled using relevant predicates to ensure appropriate counts.

Postulate 2: *As the number of known subclasses of a class increases, the influence exerted by the subclasses on each other (proportionately) decreases.* This computation is performed recursively in the object hierarchy from each primary class to the lowest common ancestor (LCA) of the primary class and target object. Equation 2 is modified as:

$$\psi_n = \begin{cases} 0 & if \quad a_n = 0 \\ \dfrac{ln(a_n) + \xi}{\prod_{h=1}^{H_n} W_h} & otherwise \end{cases} \tag{3}$$

where H_n is the height of the LCA of the target object and the primary class under consideration. For a class node on the path from the primary class to the LCA, W_h is the number of children of the node at height h; $W_1 = 1$ for primary classes because the first postulate considers object instances.

Postulate 3: *Each primary class with instances (in a room)* independently *provides support for the target's existence*

[5]Fechner's law (1860) serves as the basis of modern psychophysics.

(in the room). The evidence for the target's existence in room k is thus the summation of the evidence from N primary classes:

$$\alpha_k = \sum_{n=1}^{N_k} \psi_{n,k} = \sum_{n=1}^{N_k} \frac{ln(a_{k,n}) + \xi}{\prod_{h=1}^{H_{k,n}} W_{k,n,h}} \tag{4}$$

where N_k is the number of primary classes that have specific object instances in room k. Equation 4 thus extends the definition of ψ_n from Equation 3. The values of α_k are computed using the cardinality of the set of relevant answer set statements obtained through inference in the KB. For target localization, these three postulates (together) consider knowledge about the occurrence of specific object classes in specific types of rooms; future work may explore probabilistic models of these relationships learned from historical data.

To convert the relevant statements in the answer set into a multinomial prior that can be combined with the probabilistic POMDP beliefs, let event E_k represent the target object's existence in room k, and let E represent the target's existence in one of the rooms. Let $p^{KB}(E_k|E)$ be the probability that the target is in room k given its existence in the domain. Based on the current answer set, the entries of b^{KB}, the multinomial prior distribution over the rooms, are given by:

$$b_k^{KB} = p^{KB}(E_k|E) = \alpha_k/\alpha_0 \tag{5}$$

where $\alpha_0 = \sum_k \alpha_k$. As an example, in the simulated domain in Figure 2, let the target object be a printer that (unknown to the robot) is on the floor of the `study`. Consider a subset of the domain objects:

```
1 pillow:bedding:object, in bedroom
1 mattress:bedding:object, in bedroom
1 computer:computer-access:object, in study
1 fax:computer-access:object, in study
3 book:books-magazine:object, in study
2 magazine:book-magazine:object, in study
1 coffee_machine:kitchenware:object, in kitchen
1 fridge:kitchenware:object, in kitchen
1 book:book-magazine:object, in kitchen
1 printer:computer-access:object, unknown
```

Integers at the beginning of each line represent the number of instances of the corresponding objects. Each line also contains the relevant subset of the object hierarchy, e.g., the class `pillow` is a child of the class `bedding`, which is a child of class `object`. Let rooms in Figure 2 be indexed in ascending order from left to right. Consider α_1, the support for the target object (printer) being in bedroom ($i = 1$). There are instances of `pillow` and `mattress` in this room, so $N_1 = 2$. Since there is only one pillow known to be in the bedroom, $a_{1,1} = 1$. The LCA of the target object and class `pillow` is the root node (`object`), so $H_{1,1} = 3$. The evidence provided by sibling classes is considered in a bottom-up manner, and the extent of support is diluted as we proceed up the hierarchy, with $W_{1,1,1} = 1$. Since `bedding` and `object` have two and four children respectively, $W_{1,1,2} = 2$ and $W_{1,1,3} = 4$. The second object class with an instance in the `bedroom` is `mattress`, and $a_{1,2} = 1$ because there is only one mattress—$W_{1,2,1} = 1$, $W_{1,2,2} = 2$, and $W_{1,2,3} = 4$. The support for the printer's existence in room 1 is then computed as $\alpha_1 = 0.250$ using Equation 4. Following the same procedure, the support vector for the target object's existence in the rooms is: $\alpha = [0.250, 1.141, 0.375]$. The multinomial prior of the target's existence in the rooms is then computed (using Equation 5) as: $b^{KB} = [0.142, 0.646, 0.212]$.

2) **Computing Posterior Belief using Bayes Rule:** It is challenging to provide a Bayesian treatment for using the multinomial prior and the POMDP belief distribution to compute the posterior belief of the target's location in the domain. The KB may contain incomplete or outdated information, sensor observations are imprecise, and actions are non-deterministic. In addition, the answer set that informs the multinomial prior is subject to non-monotonic logical inference, making it difficult to use a new prior to revise the posterior computed using the previous prior. To address these challenges, the fact that actions in our domain do not change object locations is exploited to maintain the likelihood of the sequence of observations received by the robot over time. The i^{th} entry of this likelihood vector is the likelihood of the sequence of observations conditioned on s_i being the true location of the target object:

$$b_{i,t}^{Ob} = p_i(o_{1:t}|a_{1:t}, b_{i,0:t}) = \prod_{j=1}^{t} O(s_i, a_j, o_j) \tag{6}$$

Now, when an update to the KB causes a change in the answer set, Bayes rule is used to compute the revised posterior belief b' based on the multinomial prior and the likelihood of the observation sequence:

$$b'_{i,t} \propto b_{i,t}^{Ob} \cdot b_i^{KB} \tag{7}$$

This update considers the current beliefs encoded in the KB and all the observations used with the previous multinomial prior. The update is performed at the level of cells by distributing the multinomial prior for each room over the cells in the room. The revised posterior belief of the target's location is input to the VS-POMDP policy to choose an action, causing the robot to move and/or analyze an appropriate scene.

This belief update brings up an interesting, subtle and important issue about (re)use of observational information in our architecture. Each statement added to the KB corresponds to a hypothesis, based on one or more observations over a time period $(0:t)$, which has been elevated from being associated with a high probability to being associated with complete certainty. Such a *commitment* made at time t is used for inference in the ASP KB. The corresponding multinomial is then pushed back to the POMDP as the prior in Equation 7 for the Bayes rule update (say, at time $t+1$). Strictly speaking, the previous observation sequence should be discarded at this point, which can be accomplished by resetting the observation sequence likelihood to 1 when the new multinomial prior is obtained. This *observation discard* strategy, however, also discards many observations with useful information that may not have yet had a chance to support a commitment to the KB—the observation sequence typically contains far more information than was used to submit a single commitment. In addition, information about events not directly relevant to

the task may have revised the KB. Thus an *observation re-use* strategy, which does not reset the observation likelihood, allows additional inferences to be drawn later on. While this re-use is strictly incorrect in Bayesian terms, we verified experimentally that it significantly increases target localization accuracy and decreases the localization time. Thus we retain it as a *feature* of our architecture that separates logical inference from probabilistic inference. This separation is at the heart of the inferential efficiency in our architecture that avoids exact but inefficient probabilistic reasoning over the ASP KB.

D. Reasoning about Target Existence

It is possible that the object the robot is searching for does not exist in the entire domain. The robot may also have access to historical data from comparable domains that, if combined correctly with the robot's unreliable observations, can be used to estimate the probability of the target's existence in the current (search) domain. Furthermore, the robot cannot search indefinitely, but must choose when to give up the search if it cannot find the object. Intuitively, the more certain the robot is that the target exists in the domain, the longer it should persist before giving up. However, this reasoning is not captured by standard POMDP models; introducing such reasoning also negates the invariance properties used to efficiently compute the convolutional policies in our hierarchy of POMDPs. One significant contribution of this paper is a metareasoning approach to combine the historical data with domain knowledge and the current observations to terminate search appropriately. Our approach models the confidence in the historical data using a meta density over the probability that the target exists in the domain. In the derivation below, we assume that the robot has to find one instance of the target; we do not model the probability distribution over the number of instances of the target object type in the domain.

Our metareasoning approach comprises three steps: (1) using a Beta density (a meta density) to model prior knowledge from historical data and the KB about the target's existence in the domain; (2) maintaining the likelihood of the observation sequence given the existence or non-existence of the target; and (3) using the prior and the likelihood to obtain the posterior probability of target's existence in the domain. For localizing a specific object, steps 2 and 3 are repeated until the robot makes a decision about the presence or absence of the object in the domain (more details below).

The prior probability that the target exists in the current domain is $\theta = P(E)$, the parameter of a Bernoulli distribution. We therefore use a Beta probability density function (PDF) as a meta density over θ, i.e., as the conjugate prior:

$$\mathscr{B}(\theta|\alpha',\beta') = \frac{\Gamma(\alpha'+\beta')}{\Gamma(\alpha')\Gamma(\beta')}\theta^{\alpha'-1}(1-\theta^{\beta'-1}) \tag{8}$$

where the Gamma (Γ) function is used for normalization. The parameters α' and β' are (respectively) the support for existence and non-existence of the target in the domain; these parameters include the evidence from the answer set and counts of the number of times the desired object was found to exist or not exist during previous searches in other domains of the same type, e.g., other offices. The Beta PDF thus models the confidence in the combination of the knowledge of the current domain and historical data from comparable domains.

In addition to the Beta PDF, the robot computes the likelihood of the observation sequence at each time step given that the desired target object exists or does not exist in the domain:

$$\begin{aligned} p(o_t|E,a_t,b_t) =& \textstyle\sum_{\forall i\in FoV} O(o_t,a_t,s_i)\, b_t(i) \quad \text{if } o_t = o^+ \\ &+ p(FP)\cdot \textstyle\sum_{\forall i\notin FoV} b_t(i) \\ =& \textstyle\sum_{\forall i\in FoV} O(o_t,a_t,s_i)\, b_t(i) \quad \text{otherwise} \\ &+ p(TN)\cdot \textstyle\sum_{\forall i\notin FoV} b_t(i) \\ p(o_t|\neg E,a_t,b_t) =& p(FP) \qquad \text{if } o_t = o^+ \\ =& p(TN) \qquad \text{otherwise} \end{aligned} \tag{9}$$

where $p(FP)$ and $p(TN)$ are the false positive and true negative rates (respectively), obtained experimentally and encoded in the POMDP models, and FoV is the event that the target is in the robot's field of view. Since the current action (a_t) and belief (b_t) are known at each time step, they are occasionally omitted in the equations below.

Given the prior and the observation likelihood, the posterior probability of target's existence in the domain is given by:

$$p(E|o_{1:t}) = \int_\theta p_\theta(E|o_{1:t})\; p(\theta) d\theta \tag{10}$$

where $p(\theta)$ is modeled by the Beta PDF. For a given θ, Bayes rule can be used to iteratively compute:

$$p_\theta(E|o_{1:t}) = \frac{p(o_t|E)\; p_\theta(E|o_{1:t-1})}{p(o_t|E)\; p_\theta(E|o_{1:t-1}) + p(o_t|\neg E)\; p_\theta(\neg E|o_{1:t-1})} \tag{11}$$

where $p(o_t|E)$ and $p(o_t|\neg E)$, shorthand for $p(o_t|E,a_t,b_t)$ and $p(o_t|\neg E,a_t,b_t)$ respectively, are computed using Equation 9, and b_t is the result of state estimation in the VS-POMDP assuming that the object exists. The posterior can be used for early termination of the search for the target object if the probability of non-existence of the target in the domain, $p(\neg E|o_{1:t})$, exceeds a threshold (τ^-), just as the existence of the object in a specific room or cell can be confirmed when the mode of the belief (b_t) exceeds a threshold (τ^+). However, it is difficult to compute the integral in Equation 10 in closed form, and so we consider three approximations.

Expectation-based approach: The first approximation strategy computes the posterior by considering the expectation of the Beta PDF as the prior probability of existence of the target, i.e., $p(E) = \frac{\alpha'}{\alpha'+\beta'}$. The task of computing the posterior collapses to a Bayesian update, as described in Equation 11. Although it simplifies the computation of the posterior, this strategy does *not* use the Beta PDF's variance, which provides important information about the degree of belief associated with any specific $p(E)$. In other words, the meta density is effectively discarded and the estimated likelihood of existence is assumed to be correct, no matter how little or much data or knowledge the estimate is based on.

Upper-bound approach: The second strategy also considers a single value of θ from the Beta PDF as the prior probability of existence of the target object in the domain. However, this prior $\theta = p(E)$ is chosen such that $\int_0^\theta f(x)\mathrm{d}x =$

$value_{ub}$, where $f(x)$ is the Beta PDF. The motivation for this strategy is to obtain a kind of *upper bound* on the value of the prior. For instance, we use $value_{ub} = 0.9$, i.e., if the robot decides to terminate the trial for a specific target object, it would have arrived at the same decision if it had started with any of the 90% of the values of the prior θ sampled from the Beta PDF. Similar to the expectation-based approach, computing the posterior probability of existence of the target object collapses to a Bayesian update (Equation 11). However, unlike the expectation-based strategy, the Beta PDF's variance contributes to the selection of the prior probability of the target object's existence in the current domain.

Monte-Carlo sampling: The third strategy uses Monte-Carlo (MC) sampling to estimate the integral in Equation 10. In this approach, the number of samples required to approximate the integral may need to be revised during runtime to obtain a good estimate of the expected value of the posterior. To perform a belief update with the new samples, the likelihood of the observation sequence is maintained for metareasoning, similar to the approach used in Section III-C2:

$$p(o_{1:t}|E,a_{1:t},b_0) = p(o_{1:t-1}|E,a_{1:t-1},b_0)\cdot p(o_t|E,a_t,b_t)$$
$$p(o_{1:t}|\neg E,a_{1:t},b_0) = p(o_{1:t-1}|\neg E,a_{1:t-1},b_0)\cdot p(o_t|\neg E,a_t,b_t) \quad (12)$$

where $p(o_t|E,a_t,b_t)$ and $p(o_t|\neg E,a_t,b_t)$ are given by Equation 9. Note the dependence of this calculation on the sequence of actions $a_{1:t}$ and the initial VS-POMDP belief b_0. In the MC sampling strategy, the robot first draws an initial number (N_0^{mc}) of samples $\theta_i = p(E)$ from the Beta PDF, and then follows the following iterative sequence:

1. Compute observation likelihood after the standard POMDP belief update—Equation 12.
2. For each sample θ_j, compute the posterior belief, where η is a normalization term:
$$p_j(E|o_{1:t}) = \eta\cdot p(o_{1:t}|E,a_{1:t},b_0)\cdot p_j(E)$$
$$p_j(\neg E|o_{1:t}) = \eta\cdot p(o_{1:t}|\neg E,a_{1:t},b_0)\cdot (1-p_j(E)) \quad (13)$$
3. Compute the MC approximation of the posterior in Equation 10 as: $p(E|o_{1:t}) = \frac{1}{N_t^{mc}}\sum_{\forall j} p_j(E|o_{1:t})$.
4. Re-compute the number of samples needed:
$$N_t^{mc} = \{\frac{z_\sigma \cdot stdev(p_i(\neg E|o_{1:t}))}{mean(p_i(\neg E|o_{1:t})) - \tau^-}\}^2 \quad (14)$$
where the objective is to have enough samples to make a decision about early termination with a desired level of confidence ($z_\sigma = 1.645$ for 90% level of confidence).
5. If additional samples are required, draw these samples and repeat steps 2-3 above.

This strategy requires more computational effort than the other two strategies, but fully uses the variance of the Beta PDF to compute the desired posterior probability. We compare the three strategies experimentally in Section IV.

Algorithm 1 summarizes the overall control loop for belief update and metareasoning. The pre-processing step in line 1 includes, for instance, the creation of the initial VS-POMDP belief using information from the answer set, and drawing of the initial set of samples from the Beta PDF for the MC sampling strategy. In each iteration, the robot performs a POMDP belief update after executing an action and generating an observation (lines 4-6). If the KB is revised, the new answer set is used to compute a multinomial prior and thus the corresponding posterior belief distribution (lines 7-10). Next, the robot reasons about the target object's existence in the domain (lines 11-12); the optional post-processing step in line 12 includes, for instance, the creation and update of new samples for the MC sampling strategy. The search for the desired object is terminated when it is localized with high probability or the probability of its non-existence in the domain is high (lines 14-18). Although it is not shown in Algorithm 1, it is also possible to terminate the search after a fixed amount of time.

Algorithm 1: Control Loop

Input: POMDP: set of states (S), set of observations (Ω), set of actions (A), transition function (T), observation function (O), initial belief distribution (b_0), policy (π).
Input: Domain map (M), target object, robot's initial position, belief thresholds for existence (τ^+) and non-existence (τ^-) of target object.
Input: Beta PDF ($\mathscr{B}$) of prior probability of target existence in the domain.
Output: Target object *Found* (1) or *Notfound* (0).

```
1  Pre-processing step to initialize desired data structures.
2  Initialize time step t = 1.
3  while true do
4      Select an action, a_t, based on π and b_{t-1}.
5      Execute action and make an observation, o_t.
6      Perform POMDP belief update to obtain b_t.
7      if KB changed then
8          Compute new multinomial prior (Equation 5)
9          Update b_t with revised posterior belief (Equation 7).
10     end
11     Compute observation likelihoods for metareasoning
       (Equation 9).
12     Compute posterior probability of object existence in the
       domain (Equations 10-11).
13     Complete post-processing steps, if any.
14     if p(¬E|o_{1:t}) > τ^- then
15         return Notfound.
16     else if max(b_t) > τ^+ then
17         return Found.
18     else
19         t ← t + 1.
20     end
21 end
```

IV. Experimental Setup and Results

Experiments were conducted in simulated domains and on wheeled robots visually localizing target objects. The objective was to evaluate three hypotheses: (H1) representing and reasoning with default knowledge in ASP can significantly reduce the search space (and thus the time needed) to localize objects at the level of rooms; (H2) using the proposed architecture significantly increases target localization accuracy (at the level of cells in rooms) and reduces the localization time in comparison with using ASP or POMDPs individually, or using the previous approach to generate and merge ASP and POMDP beliefs [32]; and (H3) metareasoning with domain-specific

observations and historical data enables robots to reliably and efficiently determine when a trial should be terminated, and strategies that use the variance of the Beta PDF provide a good trade-off between accuracy and time. During the evaluation of hypotheses H1 and H2, metareasoning is not included. Furthermore, the experimental trials thoroughly analyze the effects of incrementally revising the KB.

Since experimental trials predominantly include the same set of axioms, and separate experiments are conducted with and without default knowledge (details below), the term "domain knowledge" is used below to primarily refer to the % of objects whose room locations are known to the robot. In each experimental trial, the ASP KB includes the hierarchy of relevant object classes and a subset of specific object instances. The robot's initial cell-level location, target object(s), and the cell-level location(s) of the target object(s), are chosen randomly; the robot does not know the location of any target object. Although this random choice makes it difficult to compute a meaningful estimate of variance in the experimental results, statistical significance is established through paired trials. In each paired trial, for each approach being compared (e.g., ASP+POMDP vs. POMDP), the initial cell-level location of the robot, the target(s), and the cell-level location(s) of the target(s), are fixed, and the robot has the same amount of domain knowledge. The robot confirms the location of an object in a grid cell when the corresponding belief exceeds a threshold ($\tau^+ = 0.80$); the threshold for claiming non-existence of the target in the entire domain (τ^-) varies between different sets of trials (details below). Unless otherwise stated, there is no time limit for an experimental trial. Target localization accuracy is considered to be maximum when the reported location and the ground truth location of an object are identical (e.g., same grid cell). The accuracy falls off as a Gaussian function of the distance between the reported location and the ground truth location.

A. Experiments in Simulated Domains

The domain used for simulation experiments extends the illustrative domain in Figure 2 (with a `bedroom`, `study`, and `kitchen`) by including one more room: `livingroom`. We use learned object models [19] and observation models to simulate motion and perception. Fifty objects in 10 different categories were simulated in these rooms, with each room comprising 25 cells. Each data point in the results described below is the average of 5000 simulated trials, and time is measured in simulation time units.

(H1) Using ASP: Figure 4 summarizes experimental results in which only the ASP KB is used to infer the target objects' locations. *ASP-based inference can only determine the room-level location of the target object*, and cannot provide the cell-level location of the object in a room. First, consider the experiments in which default knowledge is not included in the KB. In these trials, if the robot is given the room locations of all other objects (i.e., all domain knowledge), it can correctly infer the room location of the target. The accuracy decreases when the amount of domain knowledge decreases, e.g., with 40% of domain knowledge, the robot can correctly identify

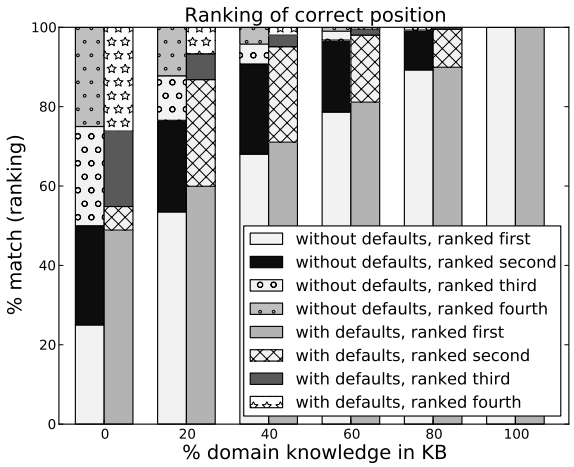


Fig. 4: Target localization accuracy using only ASP-based inference, with and without default knowledge. The correct room locations of target objects are in the top two choices in $\approx 90\%$ of the trials with as little as 40% domain knowledge; using default knowledge further improves the performance.

the target's room location with ≈ 0.7 accuracy. However, even when the locations of only 40% of the objects are known, the correct room location of any specific target object is in the top two choices in $\approx 90\%$ of the trials.

The experimental trials were repeated after including default knowledge about room locations of objects in the KB, e.g., "books are usually in the study". Although such knowledge can be useful, observations in the current domain may contradict it, e.g., someone may have left a book in the bedroom by mistake. As described in Section III-A, ASP provides good expressiveness for defaults and exceptions to defaults, and supports non-monotonic logical inference. Figure 4 shows that using default knowledge improves accuracy in comparison with the trials in which default knowledge is not used, especially when the amount of domain knowledge considered is small(er). A key outcome of ASP-based inference, especially with default knowledge, is thus the significant reduction in the search space; an indirect outcome is the reduction in the target localization time. However, ASP (by itself) is not well-suited to represent or use the probabilistic information extracted by processing sensor inputs.

(H2) Using ASP and POMDPs: The next set of experiments used both the logical inference (ASP) and probabilistic planning (POMDPs) components to localize target objects. For any given target, ASP-based inference provides a multinomial prior for POMDP state estimation, with the posterior beliefs used to determine the robot's sensing and navigation actions for localizing the target. Figure 5 summarizes the experimental results as a function of the amount of domain knowledge used to generate ASP-based beliefs; the blue-colored plot with triangular markers depicts the target localization time, and the red-colored plot with star-shaped markers depicts the target localization accuracy (measured at the cell-level). Trials corresponding to each sample point on the localization time plot were terminated when the belief in a specific cell

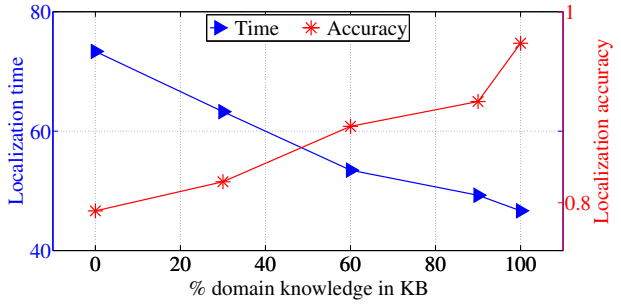


Fig. 5: Target localization accuracy and localization time as a function of the % of domain knowledge in the KB. Proposed architecture increases target localization accuracy and reduces target localization time in comparison with using only POMDPs. Paired trials establish statistical significance.

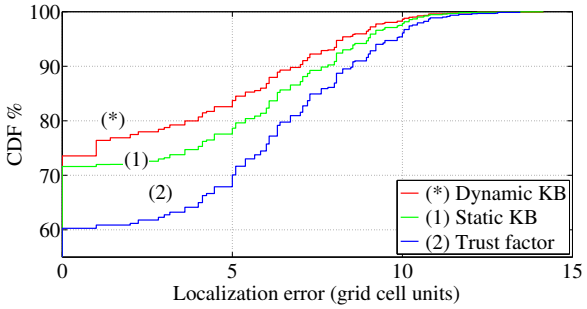


Fig. 6: Analysis of approaches for generating posterior belief, and the effects of revising the KB; using ASP inference-based multinomial priors for POMDP state estimation significantly reduces localization errors, and incrementally revising the KB further improves the accuracy.

exceeded the threshold ($\tau^+ = 0.8$); trials corresponding to the localization accuracy plot were terminated after 100 time units.

Trials corresponding to 0 on the x-axis represent the use of only the hierarchy of POMDPs—see Section III-B and [34]. The results indicate that using our proposed approach to compute the posterior belief significantly increases the target localization accuracy. Some of the localization errors are due to the room-level support for target existence provided by related objects. This problem is more pronounced when the amount of domain knowledge included in the KB is small, causing the robot to explore irrelevant locations and provide an incorrect result when the time limit is exceeded and/or some observations are incorrect; given more time, the robot is able to recover from these errors. As the robot obtains more domain knowledge, the localization accuracy steadily improves. For instance, over trials in which the robot knows the room location of all objects except the target, accuracy is 0.96 and errors are due to the target object being close to the edge of two or more cells. To establish statistical significance, we conducted paired trials; in each set of trials using just POMDPs or ASP and POMDPs, the initial cell-level locations of the robot and the target(s) were fixed, and the robot started with the same amount of domain knowledge, e.g., room locations of 40% of the domain objects. The improvement in localization time over 1000 trials (each) is significant at the 95% significance level with the p-value $< 10^{-24}$. Our architecture thus exploits the complementary strengths of logical inference and probabilistic planning to significantly reduce the localization time while also increasing the localization accuracy.

Posterior belief generation: Next, we evaluated our proposed approach for obtaining the posterior belief of the target's cell-level location using the ASP-based multinomial prior for POMDP state estimation, and analyzed the effects of incrementally revising the KB. The KB was initialized with 20% domain knowledge in each trial, and information about a few randomly chosen objects was added periodically to simulate learning from sensor inputs. Inference in the revised ASP program provides new multinomial priors for POMDP state estimation and subsequent action selection. Each trial terminates when the belief in a cell exceeds 0.8 or the time limit of 100 units is exceeded. To make the trials more challenging, some extra (Gaussian) noise was added to the observations received by the robot.

Our proposed approach for generating the posterior belief of the target's (cell) location ("dynamic KB"; Section III-C2) was compared with two approaches: (1) not revising the KB that is populated with 20% domain knowledge at the beginning of each trial ("static KB"); and (2) using relative trust factors to merge a heuristically generated ASP-based belief distribution with the POMDP belief distribution ("trust factor") [32]. The trust factor approach did not encode default knowledge, used heuristics to convert answer sets from a static KB to a belief distribution, and performed weighted averaging of this distribution and the POMDP belief distribution using the r-norm measure. To enable comparison with such an approach, trials were conducted without default knowledge in the KB, and Figure 6 summarizes the results in the form of cumulative distribution function (CDF) plots. The x-axis represents the localization error in units of grid cells, and the y-axis represents the % of trials with errors below a specific value. For instance, with our approach, $\approx 80\%$ of the trials have a localization error of ≤ 4 units, while only 66% of the trials provide similar accuracy when trust factors are used—even with a static KB, our approach results in better performance than using trust factors. Although not shown in Figure 6, using trust factors may also result in lower localization accuracy than not using ASP-based inference—based on the choice of the relative weights, it is possible for an incorrect ASP-based belief to overwhelm the POMDP belief distribution that is revised based on actual observations. Similar results were obtained in trials conducted after changing the amount of initial domain knowledge. Including default knowledge further increases the target localization accuracy and decreases the target localization time of our proposed approach. Furthermore, paired trials established the statistical significance of the performance of the proposed approach (with or without a static KB) in comparison with the trust factor approach; p-values of 3.9×10^{-69}, 1.3×10^{-9} and 2.4×10^{-30} for dynamic KB vs. trust factor, dynamic vs. static KB and static KB vs. trust factor respectively.

(H3) Metareasoning strategies: Experiments were then con-

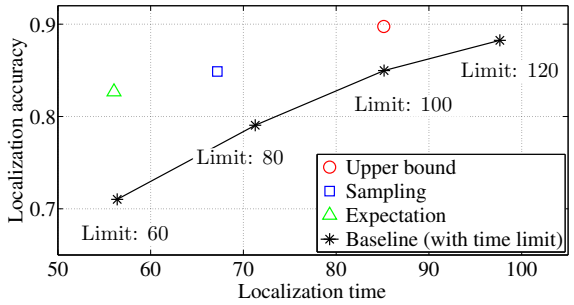


Fig. 7: Metareasoning with historical data reduces target localization time (by early termination) and (indirectly) increases the localization accuracy. The three approximation strategies trade-off between accuracy and time. Paired trials establish statistical significance of the results.

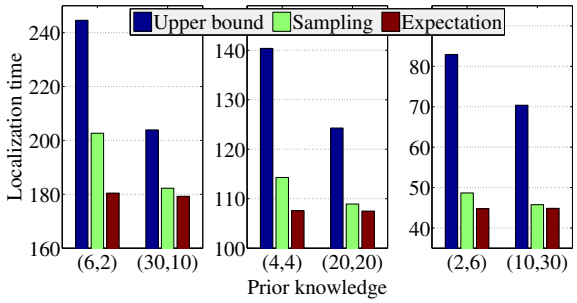


Fig. 8: Localization time as a function of prior knowledge of the target's existence in the domain. Prior knowledge is encoded as a Beta PDF—parameters (a,b) of the PDF denote the support for the existence and nonexistence of the target based on historical data from comparable domains.

ducted to evaluate the benefits of metareasoning with historical data in conjunction with observations in the current domain. To isolate the effect of using observations, the KB is static in each trial, and the target is randomly selected to be present or absent (unknown to the robot). The "baseline" strategy terminates the trial when the probability of one of the grid cells exceeds the preset threshold ($\tau^+ = 0.8$) or the trial takes longer than a given time limit. This strategy is compared with the action selection policies corresponding to the three approximation strategies ("expectation", "upper bound" and "sampling") described in Section III-D. Each of these three policies terminate a trial early if the probability of the target's non-existence in the domain exceeds a preset threshold (τ^-)—we experimented with different values of this threshold, as described below. All four policies include our proposed approach of using ASP-based multinomial prior for POMDP state estimation.

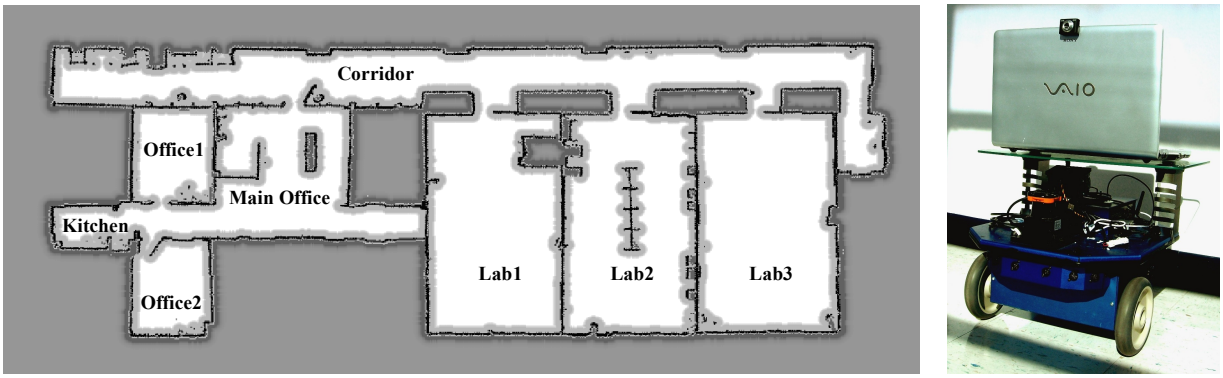
Figure 7 summarizes the results ($\tau^- = 0.7$), with the localization time and accuracy on the x-axis and y-axis respectively. The black plot with plus-shaped markers depicts the average results with the baseline strategy and specific time limits; the robot can localize the target more accurately if given more time. However, in trials in which the target object does not exist in the domain, the baseline strategy cannot terminate trials early. The action selection policies based on the three proposed approximation strategies enable early termination by updating the belief of the target's existence in the domain using historical data and observations. The results indicate that all three approximation strategies provide significantly lower target localization time in comparison with the baseline strategy; an indirect consequence is the increase in localization accuracy. For instance, to obtain a target localization accuracy of 0.85, the sampling-based strategy takes ≈ 67 time units while the baseline strategy needs ≈ 85 units. The three proposed strategies also result in different trade-offs between computational efficiency and target localization accuracy (and time). For instance, the expectation-based strategy provides the lowest localization time, but the localization accuracy is also the lowest among the approximation strategies. The upper bound strategy, on the other hand, has the highest localization time but provides the highest localization accuracy. The sampling-based strategy provides a trade-off between accuracy and time. Overall, the sampling-based and upper bound strategies result in better performance because they better exploit the variance of the Beta PDF.

To evaluate the effect of the variance of the Beta PDF, i.e., the degree of belief associated with the Bernoulli variable for the likelihood of the target's existence in the domain, the target localization time was computed using the three approximation strategies for different values of the Beta PDF's parameters. In these trials, τ^- was set to a (higher) value of 0.85 to encourage the robot to be more certain of the target's non-existence before abandoning search. As a representative example, Figure 8 summarizes the target localization time for three different sets of values of the Beta PDF's parameters. Each parameter set, e.g., $(6,2)$ and $(30,10)$, corresponds to the same expected value of the prior probability of target's existence in the domain; the variance is, however, different. Unknown to the robot, the target exists (does not exist) in the domain for 50% of the trials. The results do not differ for the expectation-based strategy that does not use the Beta PDF's variance. For each of the other two strategies, the localization time is lower within each parameter set, e.g., $(6,2)$ and $(30,10)$, if the variance associated with the prior is lower. Higher variance represents a lower degree of belief in the corresponding Bernoulli variable for the likelihood of the target's existence, and results in the robot being more conservative about terminating the trials early—the upper bound strategy has a parameter ($value_{ub}$) to control the extent to which the robot is conservative in its decisions. Furthermore, the upper bound and sampling strategies approach the expectation-based strategy in the limit of infinite historical data.

B. Experiments on a Physical Robot

Experiments were also conducted on a physical robot deployed on two floors of an office building. Figure 9(a) shows part of the map of the third floor with semantic labels assigned to specific rooms. Figure 9(b) shows the test platform—a wheeled robot equipped with cameras, laser range finder (30m, $\pm 135^o$), microphones, and an on-board computer with $4G$ RAM and 2GHz Dual Core processor.

Algorithms were implemented on the robot using the Robot Operating System (ROS) [22]. Figure 10 shows ROS nodes



(a) Domain map learned and revised by physical robot. (b) Wheeled robot platform.

Fig. 9: (a) The map of the domain, which is learned and revised by the robot using laser range data—obstacles are inflated to a distance that is based on the robot's inscribed radius. (b) The wheeled robot platform (Erratic) used for experimental trials.

corresponding to modules for path planning, localization, mapping, and acquisition of sensor data. Visual object recognition is based on learned object models that consist of appearance-based and contextual visual cues [19]. Laser range data is used by the robot to localize itself in the domain map, detect obstacles, and to determine room accessibility, e.g., if doors are open or closed. While moving between locations, the robot also periodically processes low-resolution images.

For any given task, our architecture enables the robot to perform non-monotonic logical inference in the ASP KB to provide a prior for POMDP state estimation, while POMDP planning provides a sequence of actions for visual information processing and navigation. Action execution requires the robot to move to specific locations and/or visually analyze specific scenes. The execution of each such action invokes an implementation of the corresponding algorithm in ROS, e.g., use of an existing algorithm for visual object recognition, or use of existing ROS algorithms for path planning and controlling the robot's movement. The observations obtained by executing the sensor input processing algorithms are sent to our architecture. However, not all motion goals can be achieved, e.g., a room may be inaccessible. In such situations, failure and relevant information (e.g., inaccessibility of rooms) will be reported to the ASP KB. For local path planning between two specific grid cells, we used an existing ROS path planner that builds on the A^* algorithm, and uses existing algorithms in ROS such as *trajectory rollout* and *dynamic window* for obstacle avoidance. These algorithms related to path planning have been integrated in the ROS node, *move_base*. The communication between our architecture and the path planner is achieved through the ROS *actionlib* module that provides *goal*, *feedback*, and *result* messages. Specific motion commands are sent to the platform driver by publishing to ROS topic *cmd_vel*. Figure 11 shows examples of target objects in this domain.

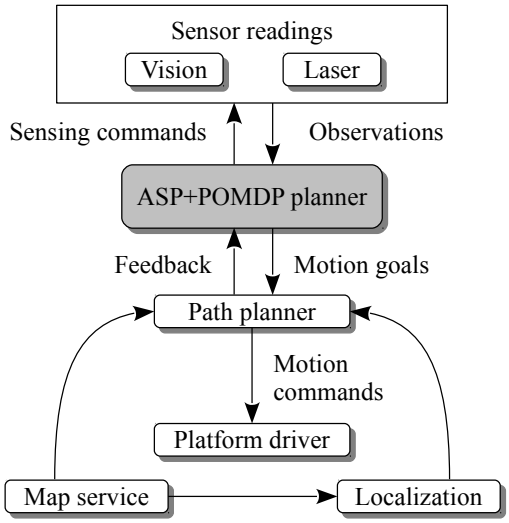


Fig. 10: Pictorial representation of a subset of nodes in the ROS implementation of our architecture.

Experimental results: We describe a representative subset of the experiment trials in which the target objects were: (1) a microwave oven; and (b) a humanoid. We compared our architecture with two baseline policies for action selection: (1) a heuristic policy that makes greedy action choices based on the current probabilistic belief; (2) a policy based on just the hierarchy of POMDPs [34]. Trials using each of the three strategies were paired, i.e., a set of trials with the three strategies used the same (randomly chosen) initial location for the robot and the same locations for the target objects. In some trials, the targets were placed in default locations, e.g., kitchen for the microwave, and lab or office for the humanoid robot. In other trials, the targets were placed in random locations. The robot does not know the ground truth locations of the target objects in any trial, but it has the learned visual models for a set of objects, a learned domain map and some domain knowledge (including default knowledge in some trials).

Table I summarizes the results of a set of 50 trials for two representative target objects (microwave and humanoid)—the results show a trend similar to that observed in the trials in simulated domains (Section IV-A). The actual target localization time can vary substantially depending on the location of the target and the initial position of the robot. We therefore report the target localization time of the two baseline strategies as a factor of the target localization time using our architecture. The results for these (and other target objects) indicate that our architecture significantly reduces the target localization

Fig. 11: Some target objects used in the experimental trials on the wheeled robot platform.

TABLE I: Target localization time of a heuristic policy, and a policy based on only POMDPs, expressed as a factor of the target localization time using our architecture. Our architecture *significantly* reduces the target localization time while successfully localizing the targets in all the trials.

Search strategies	Localization time for specific targets	
	Microwave	Humanoid
Heuristic	2.96	1.78
POMDP only	1.96	1.32
ASP+POMDP	1	1

time while successfully localizing the target objects in all the trials. For instance, the target localization time using just the POMDPs is ≈ 1.6 times, averaged across targets in addition to those considered in Table I, the target localization time using our architecture, while the factor is ≈ 2.4 for the heuristic/greedy policy. The results corresponding to the paired trials indicate that the improvement is statistically significant, e.g., p-value $\in [0.008, 0.03]$ when the target localization time obtained using our architecture is compared with that using only the hierarchy of POMDPs for localizing objects.

Representative trials: Consider two experimental trials on the mobile robot deployed in an indoor office domain. Figure 12 shows screenshots at various stages of the first experimental trial. The robot uses a learned map with known semantic labels and the target object to be localized is the humanoid observed in the last row. The screenshots capture specific steps in the sequence of actions executed by the robot as it analyzes different images of a specific subset of scenes. The robot dynamically revises the map and periodically processes images (at low resolution) as it moves between desired locations. The corresponding video is available online:
`http://youtu.be/CvKJyCI_YNE`

Consider another experimental trial to illustrate the early termination of unachievable tasks. The target object was a humanoid that (unknown to the robot) actually did not exist in the domain. Prior domain knowledge indicated that the target was likely to be in one of the two labs in the learned domain map. The robot first explored the lab that was closest: the *robot lab*. When the robot did not find the desired target after a careful visual analysis of the lab, the robot investigated the other lab. When it could not find the target object in this lab either, sufficient belief had been accumulated in favor of the target's non-existence in the domain; as described in Section III-D, the robot then terminated the trial *without investigating other rooms*. The corresponding video is available online: `http://youtu.be/2U6oOTuEd-Q`

V. Conclusions

This paper described an architecture that integrates the complementary strengths of declarative programming and probabilistic graphical models for knowledge representation and reasoning in robotics. Answer Set Prolog (ASP), a declarative language, is used to represent incomplete domain knowledge, including default knowledge that holds in all but a few exceptional situations. A hierarchy of POMDPs, an instance of probabilistic sequential decision making, is used to automatically tailor sensor input processing and navigation to tasks at hand, probabilistically modeling the associated uncertainty. An answer set obtained through non-monotonic logical inference in the ASP KB generates a multinomial prior for POMDP state estimation, using the corresponding posterior belief distribution for action selection. Inference in the KB and historical data from comparable domains are also used to generate a Beta PDF. Metareasoning with this PDF and observations enables the robot to identify eventualities not modeled by the hierarchy of POMDPs, resulting in early termination of unachievable tasks. Experimental results on a robot visually localizing objects in an office domain show that the architecture supports qualitative and quantitative representations of knowledge and uncertainty, and creates a continuous loop of non-monotonic logical inference, probabilistic planning and knowledge revision.

The architecture opens many directions for future research. First, the KB is currently not very large and uses hand-coded rules. However, ASP is capable of efficient inference in large KBs [27]—future work will scale the current approach to larger KBs, and investigate the learning of rules. We will also evaluate the architecture's capabilities for other tasks such as surveillance and reconnaissance. Second, the architecture currently only uses the inference capabilities of ASP—future work will explore the planning and diagnosis capabilities of ASP in conjunction with the probabilistic reasoning capabilities of POMDPs [33]. Third, we are investigating the integration of learning algorithms with our architecture. The long-term objective is to explore a tighter coupling between declarative programming and probabilistic graphical models for knowledge representation, reasoning, and learning, enabling the deployment of robots that can collaborate with humans in complex application domains.

Acknowledgment

The authors thank Michael Gelfond for feedback that led to revisions in the ASP formulation. This work was supported in part by the US Office of Naval Research Science of Autonomy award N00014-13-1-0766, and the European Commission-funded Strands project FP7-IST-600623. Opinions and conclusions expressed in this paper are those of the authors.

References

[1] Evgenii Balai, Michael Gelfond, and Yuanlin Zhang. Towards Answer Set Programming with Sorts. In *International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 135–147, Corunna, Spain, 2013.

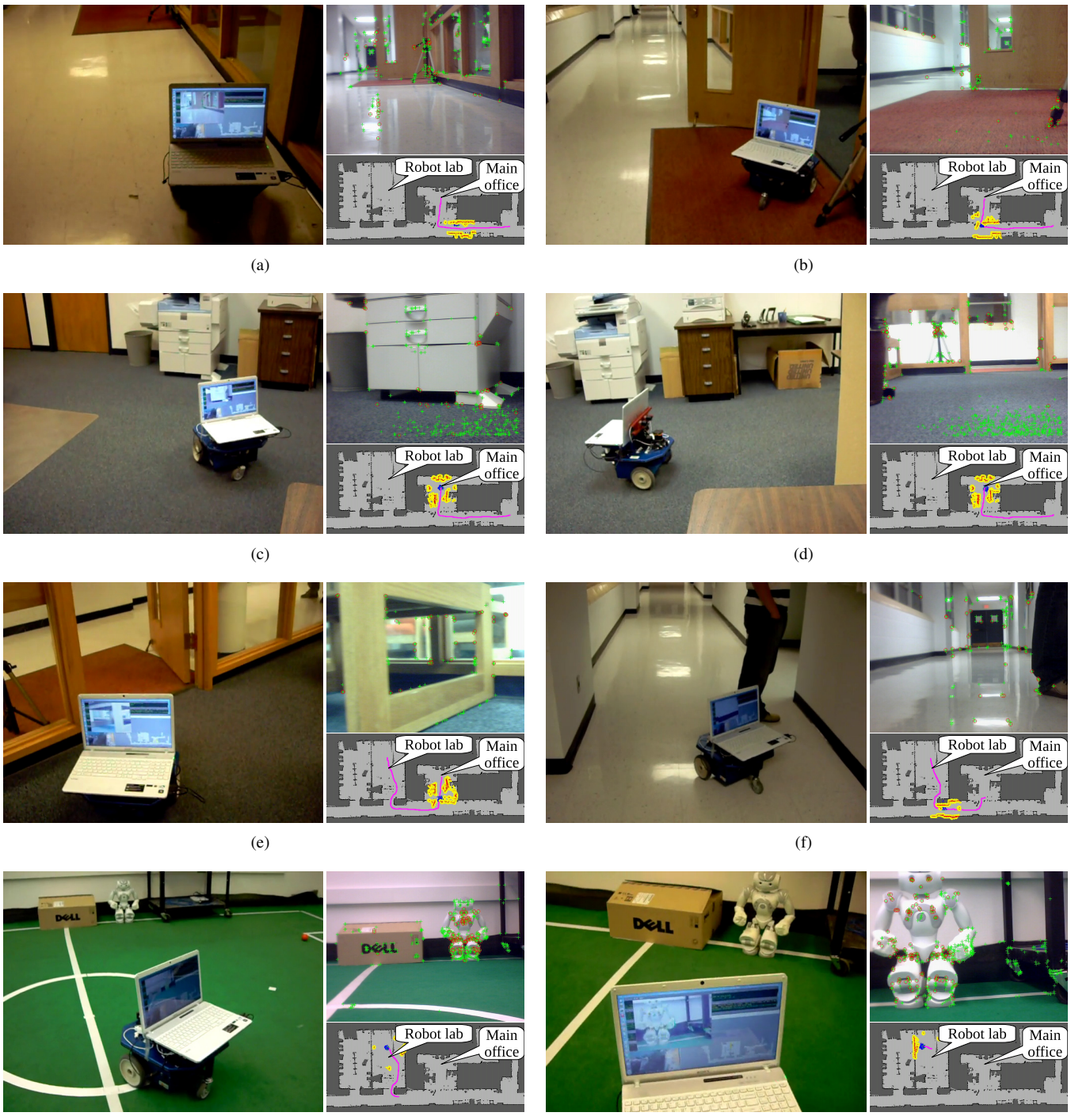


Fig. 12: Screenshots of an experimental trial in which the target is the humanoid seen in the last row. Screenshots show an external view of the robot, location of the robot in the (learned) map of the domain, and the robot's view, which lags slightly behind the robot's actual location: (a) robot starts in a map with some known semantic labels and plans a path (*pink* line) to the first location ("main office"), trading off the distance to be moved against the likelihood of finding the target; (b) the robot about to reach the first location; (c) the robot processing images of a specific scene; (d) belief update after the target was not detected in this location; (e) the robot plans a path (*pink* line) to the next (likely) target location ("robot lab"); (f) the robot avoids the obstacle (human) on its way to this location; (g) the robot analyzes images of the scene and obtains the first positive observation of the target; (h) the robot moves closer to confirm the target's presence, and to accurately localize the target. The robot dynamically revises the domain map and periodically processes images at low resolution as it moves between locations.

[2] Chitta Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.

[3] Chitta Baral, Michael Gelfond, and Nelson Rushton. Probabilistic Reasoning with Answer Sets. *Theory and Practice of Logic Programming*, 9(1):57–144, January 2009.

[4] Xiaoping Chen, Jiongkun Xie, Jianmin Ji, and Zhiqiang Sui. Toward Open Knowledge Enabling for Human-Robot Interaction. *Journal of Human-Robot Interaction*, 1(2):100–117, 2012.

[5] Roger M. Cooke. *Experts in Uncertainty: Opinion and Subjective Probability in Science*. Oxford University Press, USA, 1991.

[6] Esra Erdem, Erdi Aker, and Volkan Patoglu. Answer Set Programming for Collaborative Housekeeping Robotics: Representation, Reasoning, and Execution. *Intelligent Service Robotics*, 5(4):275–291, 2012.

[7] Esra Erdem and Volkan Patoglu. Applications of Action Languages to Cognitive Robotics. In *Correct Reasoning*, pages 229–246. Springer-Verlag, Heidelberg, Berlin, 2012.

[8] C. Galindo, J. A. Fernandez-Madrigal, J. Gonzalez, and A. Saffiotti. Robot Task Planning Using Semantic Maps. *Robotics and Autonomous Systems*, 56(11):955–966, 2008.

[9] Michael Gelfond and Yulia Kahl. *Knowledge Representation, Reasoning and the Design of Intelligent Agents*. Cambridge University Press, 2014.

[10] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, San Francisco, USA, 2004.

[11] Moritz Göbelbecker, Charles Gretton, and Richard Dearden. A Switching Planner for Combined Task and Observation Planning. In *AAAI Conference on Artificial Intelligence*, pages 964–970, 2011.

[12] Andrey Gorlin, C. R. Ramakrishnan, and Scott A. Smolka. Model Checking with Probabilistic Tabled Logic Programming. *Theory and Practice of Logic Programming*, 12(4-5):681–700, 2012.

[13] Joseph Y. Halpern. *Reasoning about Uncertainty*. MIT Press, 2003.

[14] Marc Hanheide, Moritz Gobelbecker, Graham Horn, Andrzej Pronobis, Kristoffer Sjoo, Patric Jensfelt, Charles Gretton, Richard Dearden, Miroslav Janicek, Hendrik Zender, Geert-Jan Kruijff, Nick Hawes, and Jeremy Wyatt. Robot Task Planning and Explanation in Open and Uncertain Worlds. *Artificial Intelligence*, (to appear) 2015.

[15] Marc Hanheide, Charles Gretton, Richard Dearden, Nick Hawes, Jeremy Wyatt, Andrzej Pronobis, Alper Aydemir, Moritz Gobelbecker, and Hendrik Zender. Exploiting Probabilistic Knowledge under Uncertain Sensing for Efficient Robot Behaviour. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2442–2449, 2011.

[16] Jesse Hoey, Pascal Poupart, Axel Bertoldi, Tammy Craig, Craig Boutilier, and Alex Mihailidis. Automated Handwashing Assistance for Persons with Dementia using Video and a Partially Observable Markov Decision Process. *Computer Vision and Image Understanding*, 114(5):503–519, 2010.

[17] Leslie Kaelbling and Tomas Lozano-Perez. Integrated Task and Motion Planning in Belief Space. *International Journal of Robotics Research*, 32(9-10):1194–1227, 2013.

[18] Joohyung Lee and Yi Wang. A Probabilistic Extension of the Stable Model Semantics. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford, USA, March 2015.

[19] Xiang Li and Mohan Sridharan. Move and the Robot will Learn: Vision-based Autonomous Learning of Object Models. In *International Conference on Advanced Robotics (ICAR)*, pages 1–6, Montevideo, Uruguay, 2013.

[20] Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. BLOG: Probabilistic Models with Unknown Objects. In *Statistical Relational Learning*. MIT Press, 2006.

[21] David Poole. Abducing through Negation as Failure: Stable Models within the Independent Choice Logic. *Journal of Logic Programming*, 44(1-3):5–35, 2000.

[22] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. ROS: An Open-Source Robot Operating System. In *Open Source Software Workshop*, 2009.

[23] Matthew Richardson and Pedro Domingos. Markov Logic Networks. *Machine learning*, 62(1):107–136, 2006.

[24] Stephanie Rosenthal, Manuela Veloso, and Anind Dey. Learning Accuracy and Availability of Humans who Help Mobile Robots. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 1501–1506, San Francisco, 2011.

[25] Scott Sanner and Kristian Kersting. Symbolic Dynamic Programming for First-order POMDPs. In *AAAI Conference on Artificial Intelligence*, pages 1140–1146, Atlanta, USA, July 11-15, 2010.

[26] Mohan Sridharan, Jeremy Wyatt, and Richard Dearden. Planning to See: A Hierarchical Aprroach to Planning Visual Actions on a Robot using POMDPs. *Artificial Intelligence*, 174:704–725, July 2010.

[27] Giorgio Terracina, Nicola Leone, Vincenzino Lio, and Claudio Panetta. Experimenting with Recursive Queries in Database and Logic Programming Systems. *Theory and Practice of Logic Programming*, 8(2):129–165, 2008.

[28] Nicholas Wade and Mike Swanston. *Visual Perception: An Introduction, 3rd Edition*. Psychology Press, 2013.

[29] Zeynep G. Saribatur and Esra Erdem and Volkan Patoglu. Cognitive Factories with Multiple Teams of Heterogeneous Robots: Hybrid Reasoning for Optimal Feasible Global Plans. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2923–2930, Chicago, USA, 2014.

[30] Shiqi Zhang and Mohan Sridharan. Combining Answer Set Programming and POMDPs for Knowledge Representation and Reasoning on Mobile Robots. In *Knowledge Representation and Reasoning in Robotics Workshop at ICLP*, Istanbul, Turkey, August 25, 2013.

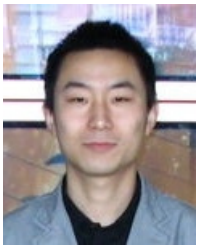
[31] Shiqi Zhang and Mohan Sridharan. Integrating Declarative Programming and Probabilistic Planning on Robots, (*Plenary talk; no reviews received*). In *AAAI Fall Symposium on How Should Intelligence be Abstracted in AI Research: MDPs, Symbolic Representations, Artificial Neural Networks, or ___?*, Arlington, USA, November 15-17, 2013.

[32] Shiqi Zhang, Mohan Sridharan, and Forrest S. Bao. ASP+POMDP: Integrating Non-monotonic Logic Programming and Probabilistic Planning on Robots. In *International Conference on Development and Learning and on Epigenetic Robotics*, pages 1–7, San Diego, USA, 2012.

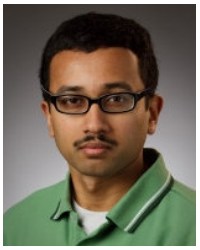
[33] Shiqi Zhang, Mohan Sridharan, Michael Gelfond, and Jeremy Wyatt. Towards an Architecture for Knowledge Representation and Reasoning in Robotics. In *International Conference on Social Robotics (ICSR)*, pages 400–410, Sydney, Australia, October 2014.

[34] Shiqi Zhang, Mohan Sridharan, and Christian Washington. Active Visual Planning for Mobile Robot Teams using Hierarchical POMDPs. *IEEE Transactions on Robotics*, 29(4):975–985, 2013.

[35] Shiqi Zhang and Peter Stone. CORPP: Commonsense Reasoning and Probabilistic Planning, as Applied to Dialog with a Mobile Robot. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 1394–1400, Austin, USA, 2015.



Shiqi Zhang is a Postdoctoral Fellow in the Department of Computer Science at The University of Texas at Austin (USA). Prior to that, he was a Postdoctoral Research Associate at Texas Tech University (USA). He has a Ph.D. in Computer Science (2013) from Texas Tech University, and a B.S. and an M.S. from Harbin Institute of Technology (China). His research primarily focuses on reasoning, sensing, and learning algorithms for intelligent robots.



Mohan Sridharan is a Senior Lecturer of Electrical and Computer Engineering at The University of Auckland (NZ), and an Adjunct Associate Professor of Mathematics and Statistics at Texas Tech University (USA). He has a Ph.D. in Electrical and Computer Engineering (2007) from The University of Texas at Austin (USA). His current research interests include knowledge representation and reasoning, stochastic machine learning, and computational vision, as applied to autonomous mobile robots and intelligent agents.

Jeremy L Wyatt is Professor of Robotics and Artificial Intelligence at the University of Birmingham (UK). He has a B.A. in Theology (Bristol), an M.Sc. in Artificial Intelligence (Sussex) and a Ph.D. in Machine Learning (Edinburgh, 1997). He has published more than 90 papers, edited three books, received two best paper awards, supervised a BCS distinguished doctoral dissertation, and led a variety of research projects. He is interested in particular in robot planning and learning in open, uncertain and unfamiliar environments.