In *Autonomous Robots and Multirobot Systems Workshop at AAMAS* (ARMS 12),
Valencia, Spain, June 5 2012.

ASP-POMDP: Integrating Non-monotonic Logical Reasoning and Probabilistic Planning on Mobile Robots

Shiqi Zhang, Forrest Sheng Bao, and Mohan Sridharan

Department of Computer Science
Texas Tech University
s.zhang@ttu.edu,forrest.bao@gmail.com,mohan.sridharan@ttu.edu

Abstract. Autonomous operation is a key challenge to the deployment of mobile robots in real-world domains such as homes and offices. The partial observability, non-determinism and unforeseen dynamic changes of these domains frequently make it difficult for robots to operate without any domain knowledge or human inputs. It is however infeasible to provide robots with all relevant domain knowledge (in advance), and humans are unlikely to have the time and expertise to provide elaborate and reliable feedback in complex domains. Our previous work enabled a team of robots to visually localize target objects using hierarchical partially observable Markov decision processes (POMDPs) [21]. Although POMDPs elegantly model the uncertainty in sensing and navigation, it is difficult to represent common sense knowledge or perform high-level reasoning with human inputs. This paper addresses these challenges by combining Answer Set Programming (ASP), a non-monotonic logic programming paradigm, with hierarchical POMDPs. Domain knowledge is represented as predicates and facts that capture the relationships between object categories, and ASP reasons with the available knowledge to initialize or revise the POMDP belief distributions. Sensory observations and human inputs cause POMDP belief updates and augment (or revise) the current knowledge modeled by ASP. All algorithms are evaluated in simulation and on mobile robots localizing targets in indoor domains.

Keywords: Hierarchical POMDP, Answer set programming, Autonomous robots.

1 Introduction

Mobile robots are increasingly being used in domains such as disaster rescue and assistive health care due to the availability of affordable high-fidelity sensors and the development of sophisticated sensory input processing algorithms. Such real-world domains are characterized by partial observability, non-deterministic action outcomes and unforeseen dynamic changes, making is difficult for a robot to process all sensory inputs using all available algorithms or to operate without any human input. At the same time, it is not feasible to provide robots will all relevant domain knowledge (in advance), and human participants are unlikely to have the time or expertise to provide elaborate and accurate feedback in complex domains.

Researchers have used partially observable Markov decision processes (POMDPs) to enable robots to robustly plan sensing, navigation and interaction (with humans) in different application domains [9, 18, 21]. It is however a challenge to include common sense knowledge obtained from sensory cues or human feedback in a standard POMDP. On the other hand, although non-monotonic logical reasoning is appropriate for knowledge representation and efficient inference, it is not well-suited for modeling the uncertainty in real-world sensing and actuation [7]. This paper presents a novel hybrid framework to address the challenges mentioned above, integrating Answer Set Programming (ASP), a non-monotonic logic programming paradigm, and our prior work on hierarchical POMDPs [21] to make the following key contributions:

- ASP enables the robot to use the information extracted from sensory (and human) cues and online repositories to robustly represent, reason with and revise spatial (and semantic) knowledge of the application domain.
- A hierarchical decomposition of POMDPs enables the robot to automatically adapt sensing and information processing to the task at hand [21]. The entropy of the POMDP belief states is used to identify the need for human feedback.
- A strategy inspired by lessons learned in psychophysics enables the robot to utilize the logical facts representing current knowledge to initialize and revise the probabilistic beliefs obtained by processing sensory inputs and human cues.

The hybrid framework and associated algorithms are evaluated in simulation and on wheeled robots that use vision as the primary source of information, in conjunction with range sensors, to localize target objects in complex indoor domains.

2 Related Work

Partially observable Markov decision processes (POMDPs) have been used to enable agents and robots to operate in domains characterized by partial observability and non-deterministic action outcomes [11]. Due to the exponential state explosion of real-world domains and the computational complexity of solvers, hierarchical and factored formulations of POMDPs have been developed for robot applications [15, 16, 21]. Our previous work used hierarchical POMDPs to enable a team of robots to localize one or more targets in indoor domains [21]. However, robots or agents operating in domains with uncertain sensing and navigation can benefit from high-level common sense reasoning using domain knowledge or human inputs [9, 13]. It is a challenge to perform such knowledge representation and reasoning in the default POMDP formulation.

Decades of research in logical reasoning and knowledge representation has provided many sophisticated algorithms [8]. Researchers are also focusing on knowledge representation and ontologies for planning perception and control tasks on robots [6, 19]. However, many of these algorithms require a significant amount of prior knowledge or are unable to robustly merge the new (uncertain) information from sensors and humans with what is currently believed by the robot based on the knowledge base. A non-monotonic logic programming paradigm such as Answer Set Programming (ASP) is well-suited for common sense knowledge representation and reasoning (especially *default reasoning*) [1, 7]. ASP has been used in many application domains and it has been integrated with natural language processing for service robots [4]. However, ASP is not well-equipped to address the uncertainty in real-world sensing and navigation.

In addition, human participants in complex domains are unlikely to have the time or expertise to provide elaborate and accurate feedback.

Recent research has resulted in the development of a *switching planner* where a robot chooses between logical reasoning or POMDPs for action selection based on expected action outcomes [9]. Such a strategy, however, fails to fully use the complementary features of logical reasoning and POMDPs. More recent research has combined deterministic and probabilistic approaches for task and motion planning [12], and combined common sense knowledge about object positions (e.g., cornflakes are typically in the kitchen) and semantic maps with topological structure for target localization [10]. However, the approach in [10] uses common sense knowledge obtained from public resources (e.g., the Internet), which may not accurately reflect the specific application and does not model non-monotonicity and dynamic changes (e.g., cornflakes are moved to living room) well. Integrating knowledge representation, logical reasoning and probabilistic reasoning therefore continues to be a formidable challenge in real-world domains. The hybrid framework described in this paper, which integrates ASP and POMDPs, is a significant step towards addressing these challenges.

3 Problem Formulation

Figure 1 presents an overview of the proposed hybrid framework. The *Knowledge Base* (KB) in ASP contains causal rules and facts about the domain. Currently, the rules are hand-coded and the facts are learned from a range of resources that include human feedback, sensory inputs and even online repositories. For any specific query (or task), reasoning in the KB results in an *answer set* containing a set of grounded literals (Section 3.1). The uncertainty in sensing and actuation is modeled using belief distributions in the POMDP (Section 3.2). The answer sets from ASP help initialize the POMDP belief or revise the existing belief distribution based on acquired knowledge (Section 3.3). The robot makes observations using sensors that are activated by action execution (e.g., visual sensing or processing algorithms) and *passive* sensors that are always in operation (e.g., range finders). The observations made with high certainty update the KB, while the remaining only update the POMDP belief distributions. Human feedback is a valuable (but limited) resource that the robot uses judiciously. For instance, if the robot knows the location of a target with high probability, it does not make sense to identify and have a conversation with a human. The question-asking strategy is hence a function of the entropy of the belief distributions (Section 3.3).

This framework is illustrated here in the context of *active target localization*, i.e., a mobile robot locating target objects in indoor domains by planning an appropriate sequence of visual sensing and information processing actions.

3.1 Knowledge Representation and Reasoning with ASP

Answer Set Programming (ASP) is a non-monotonic logic programming paradigm [1]. An ASP program is a collection of statements describing domain objects and relations between them [7]. An *answer set* is a set of ground literals that represent a set of beliefs of an agent associated with the program. Program consequences are statements that

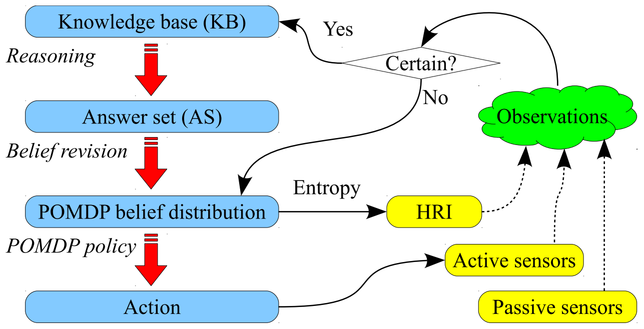

Fig. 1: Overview of the hybrid framework for knowledge representation and reasoning.

are true in all such belief sets. One key feature of ASP is its ability to perform *default reasoning*. New information can hence be used to smoothly revise statements that are currently believed to be true by the robot. The desired target localization task can hence be reduced to finding answer sets for specific queries, as described below.

Elements The semantic (2D) description of an office has the following elements:

1. Room: a space bounded by walls and doors that can be occupied by the robot and objects. The predicate `room/1` is used to define a room, e.g., `room(hallway)`.
2. Object: a visually identifiable element in a room. The predicate `object/1` defines an object, e.g., `object(fridge1)`. The desired targets are objects.
3. Category: a set of objects or sub-categories. The categories and objects are organized in the form of a tree as shown in Figure 2. Leaf nodes are objects and all other nodes are categories. Categories with objects as children are *primary categories*.

The information in the KB is used to automatically arrive at the arrangement of categories in Figure 2—it can be readily revised for other domains. Some initial information in the KB is based on online repositories (similar to [10]), which is then revised incrementally based on sensory and human inputs.

Relations between elements The following predicates represent relations between the elements. Since the domain changes dynamically (e.g., a room that was accessible may now be inaccessible), the concept of a *timestep* is introduced. The *timestep* is a natural number that increments when the robot interacts with a human or observes something with high certainty. Relations that change over time are modeled as predicates that include the timestep as a parameter.

1. `is(X,C)` implies that `C` is the parent node of `X`, where `X` is an object or a category, and `C` is a category, e.g., `is(tv, electronics)`.
2. `observed(O,R,S)` is used to create a fact when an object `O` is observed (using visual cues or human input) in room `R` at timestep `S`.

3. `located(C,R,S)` implies that objects of category `C` can be (inferred) in room `R` at timestep `S`.
4. `exclusive(C)` implies that no more than one object of primary category `C` exists in a room.
5. `accessible(R,S)` implies that the robot can enter room `R` at timestep `S`.
6. `location(R,X,Y)` states that the coordinates of room `R` are `(X,Y)`, where `X` and `Y` are natural numbers.

Reasoning rules The following rules are used for reasoning in this domain.[1]

1. If object `O` is of primary category `C`, the robot senses `O` in room `R`, and `C` is not an exclusive category, then it is believed that objects of category `C` can be in `R`.

   ```
   located(C,R,S) :- observed(O,R,S), is(O,C), not exclusive(C).
   ```

2. If objects of category `C` can be in room `R`, then the objects of the parent category of `C` can be in room `R`. Likewise, objects in all categories that are ancestors of `C` can be in `R`.

   ```
   located(C1,R,S) :- located(C2,R,S), is(C2,C1).
   ```

3. (*Rule of inertia*) An object retains its location (i.e., it exists in a room) until it is known to be in some other location; a room remains accessible (inaccessible) until it becomes inaccessible (accessible).

   ```
   observed(O,R1,S+1):- observed(O,R1,S), not observed(O,R2,S+1),
                        R1 != R2.

   accessible(R,S+1) :- accessible(R,S), not ¬ accessible(R,S+1).
   ```

Reasoning example As an illustrative example of non-monotonic reasoning in ASP, consider the following:

1. *Test-case 1* has the following facts:

   ```
   step(1..2).
   observed(printer1, lab, 1).
   is(printer1, printer).
   ```

   Reasoning in ASP produces the following answer set (existing facts not repeated):

   ```
   observed(printer1, lab, 2).
   located(printer, lab, 1).
   ```

2. Consider *Test-case 2* that has a new fact about an object's observed location:

   ```
   step(1..2).
   observed(printer1, lab, 1).
   is(printer1, printer).
   observed(printer1, office, 2).  % new fact
   ```

[1] Variable definitions: `#domain step(S)`,`#domain object(O)`, `#domain category(C;C1;C2)`, and `#domain room(R1;R2)`.

Reasoning in ASP now produces the following modified answer set (once again, existing facts not listed):

```
located(printer, lab, 1).
located(printer, office, 2).
```

Baral [1] provides a more detailed description of inference in ASP. In this paper, ASP's ability to elegantly perform non-monotonic logical reasoning is integrated with the uncertainty modeling capability of POMDPs described below.

3.2 Uncertainty Modeling with POMDP

Consider the situation where a robot has learned a domain map and has to locate a specific target. Assume that ASP has provided a set of likely locations (e.g., two rooms). To localize the target, the robot has to move and analyze a sequence of images of a sequence of scenes. Towards this objective, we use hierarchical POMDPs to account for the uncertainty in sensing and navigation. The 3D area to be analyzed is represented as a discrete 2D *occupancy grid*, with each grid storing the probability of target occurrence. The hierarchical POMDP has $2-3$ levels. The high-level POMDP poses sensing as the task of maximizing information gain, i.e., processing a sequence of scenes to reduce entropy in the probability distribution over the grid. The lower levels of the POMDP plan the processing of specific regions of images of a chosen scene using a subset of available algorithms. This work builds on our previous work [21] that enables automatic belief propagation and model creation—we briefly summarize existing work and our contributions (for completeness), focusing primarily on the high-level POMDP.

Sensing POMDP For a grid with N cells, the high-level (HL)-POMDP is the tuple $\langle S, A, T, Z, O, R\rangle$ defined as:

- $S : \{s_i,\ i \in [1,N]\}$ is the state vector; s_i corresponds to the event that the target is in grid cell i.
- $A : \{a_i, i \in [1,N]\}$ is the set of actions. Executing a_i causes the robot to move to and analyze grid cell i.
- $T : S \times A \times S' \to [0,1]$ is the state transition function; an identity matrix when actions do not change state.
- $Z : \{\text{present, absent}\}$ is the observation set that indicates if the target is detected.
- $O : S \times A \times Z \to [0,1]$ is the observation function.
- $R : S \times A \to \mathbb{R}$ is the reward specification.

The robot maintains a *belief state*, a probability distribution over the states—with no prior knowledge, the belief state is a uniform distribution resulting in maximum entropy. The HL-POMDP seeks to choose actions that significantly reduce the entropy by causing the belief distribution to converge to likely target locations. The reward of action a_t at time t is hence defined as the reduction in entropy between belief state B_{t-1} and the resultant belief state B_t. The observation function models the probability of target detection as a function of the robot position and target position, where the probability of a specific observation in cell i given that the target is in cell j and the focus is on cell k, i.e., $p(z_i|s_j, a_k)$, is a Gaussian distribution—the mean depends on target location, grid cell being examined and field of view, while the variance represents the sensitivity of sensory inputs to the object's distance from the sensor. Given these model parameters, POMDP solvers compute a *policy* that maps belief states to actions: $\pi : B_t \mapsto a_{t+1}$. Policy gradient methods are used to compute the policy that minimizes entropy over a planning horizon. The policy is in the form of a matrix of "weights", based on which an action is selected for each belief state [3].

The number of grid cells can increase exponentially in real-world domains, making real-time solutions of POMDP formulations intractable even with sophisticated solvers. This challenge is addressed based on the observation that if the robot is analyzing a specific grid cell, its performance is mainly a function of (and can affect) only a small number of surrounding cells. The robot hence learns a *policy kernel* from a baseline policy for a local region with a small number of grid cells. The policy for a larger area with a larger number of grid cells is generated by an inexpensive convolution operation. For instance, given the baseline policy generated for a 5×5 map (with 25 states and actions), the matrix of weights is reorganized into layers (each layer corresponds to a specific state) and 3×3 mask is convolved across the layers (and normalized) to generate the 3×3 kernel. This policy kernel can then be convolved with (say) a 10×10 map to generate the corresponding policy (one layer at a time). Although it may take some time for the robot to learn a baseline policy, the extracted kernel is a function of the sensors and is typically computed once—the kernel can then be used for different larger-sized domain maps.

Motion Cost and Path Planning A mobile robot has to physically move between grid cells—actuation expends time and effort, and introduces errors that accumulate as the distance traveled increases. The movement is hence associated with a cost proportional to the distance to be moved. This cost revises action weights during policy execution:

$$\hat{w}(a_j) = f(w, d_{A^*}) = w(a_j) \frac{1}{1 + \frac{d_{A^*}(a_i, a_j)}{\text{speed}}} \tag{1}$$

where $w(a_j)$ and $\hat{w}(a_j)$ are the weights corresponding to action a_j before and after the revision, while $d_{A^*}(a_i, a_j)$ is the distance between the current grid cell and the candidate grid cell. Unlike simulated domains, real-world domain maps include objects and obstacles (e.g., doors and walls), a subset of which can move (or change) over time. We enable the robot to continuously update the domain map using laser-based SLAM algorithms. We also compute the distance between cells using the A^* search algorithm. This policy revision thus enables the robot to robustly (and efficiently) account for unforeseen domain map changes and trade off the likelihood of locating the target in a grid cell against the cost of traveling to that grid cell.

We also introduce hill-climbing to make target localization in large maps more efficient. Instead of choosing the best (next) action based on current beliefs, the robot considers a path through a set of grid cells that are likely target locations. It is however infeasible to estimate an optimal path by evaluating all possible paths through all grid cells. The robot is hence enabled to first detect local maxima grid cells whose weights are much larger than surrounding cells—these local maxima are found by hill-climbing based on a small number of randomly selected initial seed points. To evaluate paths through different combinations of these local maxima grid cells, the robot computes the weighted cost of each path:

$$w^{path}([h_0, h_1, \ldots, h_N]) = \sum_{i=1}^{N} f(w(h_i), \sum_{j=1}^{i} d_{A^*}(h_{j-1}, h_j)) \tag{2}$$

where, h_n is the nth local maxima, h_0 is the current position of the robot, and $w(h_i)$ is the (action) weight at the grid-cell corresponding to hot-spot h_i. Function f is described in Equation 1. Intuitively, the path-planning strategy makes the robot focus on the *most interesting* grid cells, i.e., cells that have a high likelihood of containing the target and have other similar grid cells close by. Note that the robot does not necessarily go through the entire path—the observation obtained at the first grid cell along the path will update the belief distribution and revise the path.

For each chosen scene, the lower-levels of the POMDP plan a sequence of visual processing algorithms to process a suitable sequence of salient regions of interest in specific images of the scene—this process is based on existing work [21].

3.3 Integrating ASP and POMDP

As described in Section 3.1, the ASP-based formulation elegantly models domain knowledge and uses non-monotonic logical reasoning to robustly merge acquired information that augments (or reduces belief in) the existing knowledge. The POMDP formulation (described above) models the uncertainty in sensing and navigation to adapt sensing and processing to any given task. This section describes the integration of ASP and POMDP (Figure 1), enabling the robot to (a) use current knowledge to initialize or revise the POMDP beliefs; (b) use belief entropy to identify the need for high-level human input; and (c) merge the information extracted from sensory cues and human feedback with the existing knowledge base (KB).

Belief revision based on answer set As stated in Section 1, it is a challenge to represent common sense knowledge in POMDPs, since the acquired information may have varying levels of relevance to current and future tasks. Our framework addresses this challenge by using the current knowledge to initialize and revise POMDP beliefs.

Answer sets, as described earlier, represent all the knowledge that is currently believed to be true by the robot. The answer sets are therefore used to assign a *bias* to the initial belief distribution used by the POMDP to localize specific targets, significantly improving the efficiency of task performance. Specifically, the *literals* in the answer set relevant to the current task are used to compute an initial bias distribution. This operation (more details below) transforms the answer set into a distribution of the same form as the POMDP belief. Since the KB (and hence the answer set) can contain incomplete and/or out-of-date information, the answer set-based belief and POMDP belief are merged using relative *trust factors* (more details below). The belief initialization and revision algorithm models the following hypotheses that capture the occurrence (and co-occurrence) relationships between objects, based on the description of object categories in the knowledge base (see, for instance, Figure 2):

1. An object is more likely to be co-located with *close* "relatives", where closeness is defined as the distance to the lowest common ancestor in the tree of object categories. E.g., in Figure 2, a printer is more likely to be co-located with scanners than DVD players.
2. For any category, the influence of "siblings", i.e., of categories with a common parent, increases as the number of "siblings" decreases. The influence of a "sibling" category increases when there is sufficient support for the sibling's existence (predicate `observed/3`).
3. The relationship between the probability of object occurrence (as an entry in a belief vector) and the magnitude of evidence provided by different categories (and siblings) follows **Fechner's law**[2].

These hypotheses enable smooth and robust evidence propagation. First consider the belief generation based on the current answer set, described (for ease of explanation) in the context of target localization in a set of rooms:

$$b_i^A = \alpha \ln \left(1 + \sum_{m=1}^{M_i} \frac{N_{i,m}^F}{\prod_{k=0}^{K_{i,m}-1} N_{i,k,m}^S} \right) \tag{3}$$

where b_i^A, the probability that the target is in room i based on the answer set, is a logarithmic function (inspired by Fechner's law) of the evidence obtained from the (current) known answer set, and α is a normalizer. In Equation 3, m is the index of primary category C_m, ranging from 1 to the total number of primary categories with leaf objects known to be in room i (i.e., M_i), while $N_{i,m}^F$ counts the number of objects of C_m known in room i. Values of M_i and $N_{i,m}^F$ are calculated by counting the number of relevant `located/3` and `observed/3` literals (respectively) in the answer set. The term $K_{i,m}$ is the height (in the object category tree) of the lowest common ancestor of the (desired) target and C_m. The product in the denominator accounts for the category nodes along the path from C_m to the lowest common ancestor. The variable k, which represents the height of the nodes along the path considered, ranges from the object level (i.e., 0) to $K_{i,m} - 1$, one level lower than the lowest common ancestor. Finally, $N_{i,k,m}^S$ is the number of siblings of the node (including itself) on the path at height k—the value of $N_{i,0,m}^S$ (at height $= 0$) is set to be 1.

Once a belief vector has been computed (for target object occurrence) based on the answer set, the revised POMDP belief is obtained by merging the existing POMDP belief distribution with the belief distribution obtained from ASP:

$$b_i' = (1 - \Omega) b_i + \Omega b_i^A \tag{4}$$

where b_i and b_i' represent the sum of beliefs (that the target is in that room i) before and after the belief revision respectively. The parameter $\Omega \in [0,1]$ represents the relative trust in the knowledge represented by the answer set. The effect of this parameter is analyzed experimentally in Section 4. This is a simplistic approach to merge probability distributions—future work will consider more sophisticated algorithms [2, 5].

[2] Fechner's law was introduced in 1860 and served as the basis of modern Psychophysics. It states that subjective sensation is proportional to the logarithm of stimulus intensity.

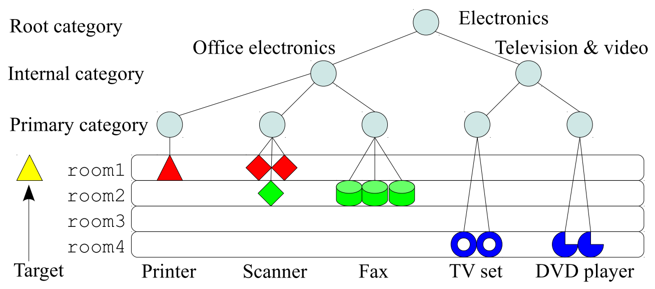

Fig. 2: Pictorial representation of relationships between object categories that are used for ASP-based belief generation.

As an illustrative example, consider Figure 2, which is a pictorial representation of a subset of object categories in the KB. This “tree” represents the following *elements*, *relations* and *learned facts* in KB—the knowledge is a combination of initial information (extracted automatically from online stores, e.g., *Amazon*) and information extracted from sensory inputs and human inputs:

- **Facts (elements):**

```
category(electronics).
category(office_electronics).
...
category(printer).
...
object(printer1).
...
room(room1). ... room(room4).
```

- **Facts (relations):**

```
is(office_electronics, electronics).
is(television_and_video, electronics).
...
is(fax, office_electronics).
...
is(printer1, printer).
...
is(dvd_player2, dvd_player).
```

- **Facts (learned):**

```
observed(printer2, room1, 1).
...
observed(scanner3, room2, 1).
...
observed(dvd_player2, room4, 1).
accessible(room1, 1).
...
location(room1, 1000, 1000).
...
location(room4, 4000, 4000).
```

In this example, there are 4 rooms with some objects classified under "electronics". The robot (at this point) knows the location of some objects, e.g., `room1` has one printer (shown as red triangle in Figure 2) and two scanners (shown as red diamonds). The robot is asked to localize a specific printer (shown as yellow triangle) that is visually distinguishable from other objects. Unknown to the robot, the object is in `room1`. ASP reasons with the existing facts to provide an answer set with the following literals:

```
located(printer, room1, 1).
located(scanner, room1, 1).
located(office_electronics, room1, 1).
located(electronics, room1, 1).
...
located(dvd_player, room4, 1).
located(television_and_video, room4, 1).
located(electronics, room4, 1).
```

The answer set is used to compute the ASP-based belief distribution (Equation 3) resulting in $b^A = [0.3890, 0.3361, 0.0000, 0.2749]$. The initial POMDP belief distribution (uniform in the absence of knowledge) is then revised as described in Equation 4, with the trust factor Ω set such that POMDP and ASP modules are trusted equally. The revised belief vector is $[0.3195, 0.2931, 0.1250, 0.2625]$. The belief for each room is spread over the grid cells in the room using a large-variance Gaussian centered in the middle of the room to motivate the robot to move to a central location in rooms. Prior knowledge about likely locations of target within a room will suitably revise mean and variance of the Gaussian. The updated beliefs are used in the learned HL-POMDP policy to choose an action, resulting in the robot moving to analyze a specific scene.

Knowledge acquisition and observations The final component is the knowledge acquisition from sensory inputs and humans. Since human participants may not have the expertise or time to provide elaborate and accurate feedback, human feedback is limited to simplistic high-level verbal inputs.

As the robot moves in the domain, (existing) algorithms are used to periodically process images to detect humans (specific humans are *not* modeled separately). When a human is detected nearby, the robot computes the need for human feedback based on entropy of the belief distribution (for the target being localized). A low entropy implies that the robot is confident of the target object's location—the human is then ignored (except for safe navigation). If the entropy is high, the robot will stop and draw the human's attention (emit tone), followed by a specific query about a room's accessibility or the target object's location. These queries (generated by robots) and valid responses are based on known templates such as:

```
Robot:  Where is the [object]?
Human:  In [room]. / Sorry, I do not know.
Robot:  Is [room] accessible?
Human:  Yes. / No. / Sorry, I do not know.
```

In addition to human inputs, the robot also acquires knowledge by processing sensory inputs. As the robot moves between grids, learned models (more details below) are used to detect objects in low-resolution images at low frequency. A newly detected object with high certainty is added to the knowledge base, using the detected position to form a suitable fact. Although this piece of information may not be directly relevant to the current task, it may still be relevant to future tasks.

Dynamic changes in indoor domains include changes in object configuration and obstacle locations, e.g., a door that was open may now be closed. The robot can confirm such changes using human feedback—changes detected with high certainty update the KB. Such revisions to the KB may eliminate certain areas from analysis in the near future. Subsequent observations can produce facts that (once again) revise the KB.

4 Experimental Evaluation

The hybrid framework was evaluated in the context of a mobile robot localizing target objects in indoor domains (e.g., multiple floors of our department). Experiments were designed to evaluate the ability of the robot to: (a) represent and reason with the knowledge acquired by sensing and from humans; and (b) make use of the knowledge to reliably and efficiently perform the desired tasks in complex real-world domains. Experimental trials were conducted in simulation and on physical (wheeled) robots to evaluate the following hypotheses: (I) the integration of ASP and POMDP enables reliable target localization while significantly reducing the time required to locate objects; and (II) the entropy-based strategy for interacting with humans enables the robot to make best use of human feedback to reliably and efficiently localize targets.

4.1 Initial Setup

The robots used in the experiments were equipped with the following capabilities not described in this paper: (a) simultaneous localization and mapping (SLAM) using laser range data; (b) local obstacle avoidance using range information; and (c) speech recognition using the Sphinx toolkit [20]. The robot learned the domain map and revised it during experiments. The robot also had an algorithm to autonomously learn sophisticated object models using a variety of visual cues [14]. In the experiments below, the object models required to recognize the desired objects have already been learned. The object models enable the robot to generate the model parameters for the lower levels of the POMDP hierarchy. The learned object models also provide certainty measures for the observations extracted from sensory inputs. In the experiments below, the robot associates high certainty with all human inputs.

4.2 Experiments in Simulated Domains

As conducting many trials on physical robots is a challenge, a simulated domain was designed for extensive evaluation. The learned object (and error) models were used to simulate realistic motion and perception. Figure 3 shows an instance where four rooms are connected by a surrounding hallway in a 15×15 grid. Table 1 shows the corresponding 50 objects in 10 different primary categories. For simplicity, we assume that none of the primary categories included in the experiments are `exclusive`. As stated

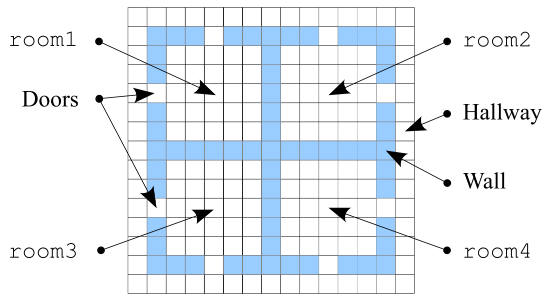

Fig. 3: Grid map of a domain created in simulation.

Table 1: Object categories for the simulated domain in Figure 3.

Categories	Root level	1									
	Internal level	1.1				1.2			1.3		
	Primary level	1.1.1	1.1.2	1.1.3	1.1.4	1.2.5	1.2.6	1.2.7	1.3.8	1.3.9	1.3.10
Object numbers	`room1`	3	2	3	1						
	`room2`				3	4	2	1			
	`room3`	1	1	7					6	1	2
	`room4`					2	3	2	1	3	2

earlier, the object category tree is created automatically using learned and known facts (and reasoned answer sets). Each data point in figures described below is the average of results obtained over 5000 trials in the simulated domain.

To evaluate hypothesis I, all 50 objects are stationary and one or more of them are randomly selected as target objects, whose locations are hence unknown to the robot. The robot's initial position is randomly selected in each trial. A specific target object is considered to be localized when one of the entries in the corresponding belief distribution exceeds a threshold (e.g., 0.9) though performance is robust to the choice of threshold (≥ 0.75). Figure 4 summarizes the experimental results, with the x-axis depicting the extent to which ASP-based beliefs are trusted. When ASP-based beliefs are not included (0 on the x-axis), the accuracy is high (≈ 0.95)—even the few errors correspond to objects in a grid cell being localized in one of the neighboring cells. However, the robot travels a significant distance (and hence spends a considerable amount of time) to locate the targets—*there is no timed termination of these trials*. As the robot starts considering the ASP-based beliefs using Equations 3-4 there is a marked decrease in the distance traveled by the robot to localize targets, and the performance is robust over a wide range of trust factors. However, when ASP-based beliefs are trusted significantly more than the POMDP beliefs (that are based on sensing), accuracy starts falling. One reason for this decrease in accuracy is that the initial knowledge about categories (extracted from online repositories) is not fully correct. Another reason is that the evidence from "related" objects can sometimes overwhelm other facts. For instance, when the object of category 1.1.1 in `room3` is selected as a target, `room1` has the highest initial belief based on the answer set. It is a challenge for the robot to recover from such situations if it mainly trusts ASP-based beliefs, especially when this trust is combined with false positive observations of the target. In other words, logical reasoning (with KB) and probabilistic reasoning are equally important for reliable and efficient operation.

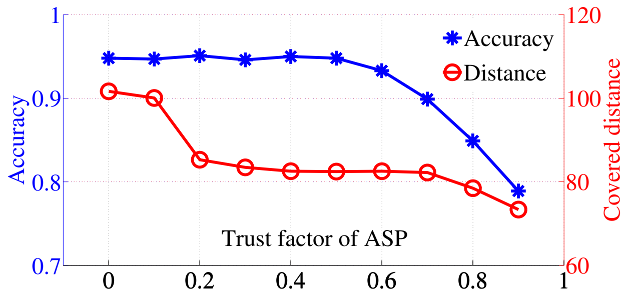

Fig. 4: ASP+POMDP: the time taken to localize targets is significantly reduced by using ASP-based beliefs, and the high target localization accuracy provided by POMDPs is retained. Trusting ASP beliefs too much has a detrimental effect on accuracy.

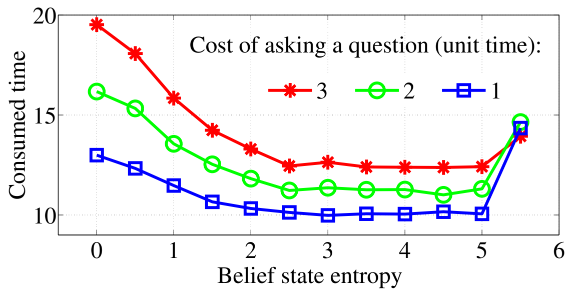

Fig. 5: Human feedback as a function of belief *entropy*. Judicious use of human feedback significantly reduces the time taken to localize targets. Frequent use of feedback hurts performance because of the uncertainty in human inputs.

Next, to evaluate hypothesis II, human feedback is considered in addition to sensory inputs. The system uses known ground truth to simulate human feedback that is *available* to the robot approximately once every five actions. In addition, there is a 20% likelihood of the feedback being incorrect. The results summarized in Figure 5 are for the domain in Figure 3, with the three curves corresponding to different costs associated with human feedback (in units of time). The trust factor for ASP is chosen in the range ($\approx 0.2 - 0.6$) that results in good performance in Figure 4. The x-axis shows the belief entropy threshold above which the robot seeks human input. When the threshold equals the maximum entropy (≈ 5.4), the robot never asks for human feedback, and the robot always seeks human feedback (when available) when the threshold is 0. Since human feedback can be unreliable, extensively seeking human input hurts performance. For any entropy threshold between $2.5 - 5.0$, the robot requires the least amount of time to localize the corresponding targets. Humans can help identify the room containing the target (*but not the exact location*) and/or comment on accessibility of rooms (see Section 3.3). Human feedback therefore helps significantly if used when needed (i.e., when the entropy is reasonably high) to reduce the search space of the robot. At the same time, if the robot rarely seeks human feedback (high threshold), target localization takes more time. In addition, as the cost of interacting with humans increases, feedback should be acquired more sparingly.

4.3 Experiments on Physical Robots

Experiments were also conducted over two floors of our department building. The third floor, for instance, has three research labs, 13 faculty offices, a conference room and a kitchen (and common area). The experimental platform was a wheeled robot running the Robot Operating System [17]—see inset in Figure 6.

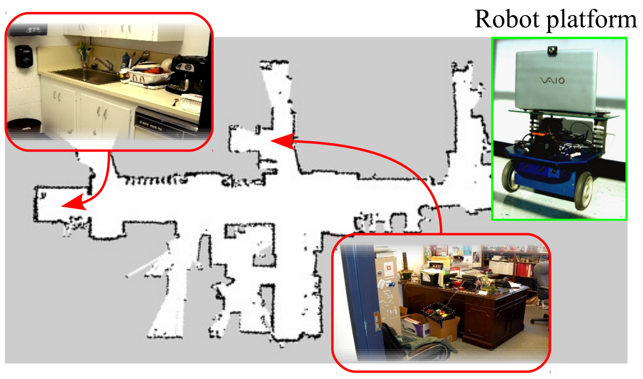

Fig. 6: Occupancy-grid map of the indoor office domain used for experimental evaluation. Robot platform shown as inset.

We conducted 30 trials (all successful) with the robot starting at a random location and given a random target, e.g., a coffee maker or a specific printer. As stated earlier, the robot has learned a domain map and object models. The robot starts with some domain knowledge and incrementally augments it. Consider a trial where the robot knows the presence of a refrigerator and a microwave in one of the rooms even though it has *no* knowledge about the semantic meaning of “kitchen”. Based on the object category tree corresponding to the current knowledge, the robot concludes that the coffee maker is highly likely to occur in the same room with other kitchenware, resulting in high initial belief (of target occurrence) in the kitchen after merging ASP and POMDP beliefs. As the robot moves to the kitchen, it meets a human but does not ask for input because the belief entropy is not high. However, in the main office outside the kitchen, the robot detects a printer that had recently been moved from the floor above, and the door to an instructor's office that is closed. These pieces of information, though not relevant to the current task, revise the KB. On arrival at the kitchen, the hierarchical POMDP enables the robot to efficiently process the images and localize the coffee maker. If the robot has to enter the instructor's office or find the printer in subsequent trials, it uses the existing knowledge and seeks human input appropriately.

5 Conclusions and Future Work

This paper presented a novel hybrid framework that integrates ASP and POMDPs to enable a mobile robot to reason with domain knowledge, automatically tailor sensing and information processing to the task at hand, merge logical facts with probabilistic beliefs, and use the information extracted from sensory cues and high-level human inputs to revise the knowledge base. The framework is evaluated in the context of visual target localization to show that the robot is able to operate reliably and efficiently.

The proposed framework opens up many directions of future research. Future work will consider advanced logical reasoning of object (and domain) properties, e.g., *kitchen typically has only one microwave*. We also seek to integrate decision-theoretic inference with probabilistic non-monotonic logical reasoning. A mobile robot will then be able to jointly reason with logical facts and model the uncertainty in sensory cues and human inputs, resulting in widespread deployment in real-world application domains.

References

1. Chitta Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2003.
2. Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2008.
3. O. Buffet and D. Aberdeen. The Factored Policy-Gradient Planner. *Artificial Intelligence*, 173(5-6):722–747, 2009.
4. Xiaoping Chen, Jianmin Ji, Jiehui Jiang, Guoqiang Jin, Feng Wang, and Jiongkun Xie. Developing High-Level Cognitive Functions for Service Robots. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Toronto, May 2010.
5. Robert T. Clemen and Robert L. Winkler. Combining Probability Distributions from Experts in Risk Analysis. *Risk Analysis*, 19(2):187–203, 1999.
6. 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.
7. Michael Gelfond. Answer Sets. In Frank van Harmelen and Vladimir Lifschitz and Bruce Porter, editor, *Handbook of Knowledge Representation*, pages 285–316. Elsevier Science, 2008.
8. Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, San Francisco, USA, 2004.
9. Moritz Göbelbecker, Charles Gretton, and Richard Dearden. A Switching Planner for Combined Task and Observation Planning. In *International Conference on Artificial Intelligence (AAAI)*, 2011.

10. 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*, 2011.
11. Leslie Kaelbling, Michael Littman, and Anthony Cassandra. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101:99–134, 1998.
12. Leslie Kaelbling and Tomas Lozano-Perez. Domain and Plan Representation for Task and Motion Planning in Uncertain Domains. In *IROS 2011 Knowledge Representation for Autonomous Robots Workshop*, 2011.
13. W. Bradley Knox and Peter Stone. Combining Manual Feedback with Subsequent MDP Reward Signals for Reinforcement Learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2010.
14. Xiang Li, Mohan Sridharan, and Shiqi Zhang. Autonomous Learning of Vision-based Layered Object Models on Mobile Robots. In *International Conference on Robotics and Automation*, Shanghai, China, May 9-13 2011.
15. S. C. Ong, S. W. Png, D. Hsu, and W. S. Lee. Planning Under Uncertainty for Robotic Tasks with Mixed Observability. *International Journal of Robotics Research*, 29(8):1053–1068, July 2010.
16. J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun. Towards Robotic Assistants in Nursing Homes: Challenges and Results. *Robotics and Autonomous Systems, Special Issue on Socially Interactive Robots*, 42(3-4):271–281, 2003.
17. 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 *ICRA Open Source Software Workshop*, 2009.
18. Stephanie Rosenthal, Manuela Veloso, and Anind Dey. Learning Accuracy and Availability of Humans who Help Mobile Robots. In *International Conference on Artificial Intelligence (AAAI)*, San Francisco, 2011.
19. Karthik Varadarajan and Marcus Vincze. Ontological Knowledge Management Framework for Grasping and Manipulation. In *IROS-2011 Workshop on Knowledge Representation for Autonomous Robots*, September 2011.
20. Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, and Joe Woelfel. Sphinx-4: A Flexible Open Source Framework for Speech Recognition. Technical report, Sun Microsystems SMLI TR-2004-139, November 2004.
21. Shiqi Zhang and Mohan Sridharan. Active visual sensing and collaboration on mobile robots using hierarchical POMDPs. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2012.