How Should Intelligence Be Abstracted in AI Research ...
AAAI Technical Report FS-13-02

Integrating Declarative Programming and Probabilistic Planning for Robots

Shiqi Zhang and Mohan Sridharan
Department of Computer Science, Texas Tech University
Lubbock, TX 79409, USA
shiqi.zhang6@gmail.com, mohan.sridharan@ttu.edu

Abstract

Mobile robots deployed in complex real-world domains typically find it difficult to process all sensor inputs or operate without substantial domain knowledge. At the same time, humans may not have the time and expertise to provide elaborate and accurate knowledge or feedback. The architecture described in this paper combines declarative programming and probabilistic sequential decision-making to address these challenges. Specifically, Answer Set Programming (ASP), a declarative programming paradigm, is combined with hierarchical partially observable Markov decision processes (POMDPs), enabling robots to: (a) represent and reason with incomplete domain knowledge, revising existing knowledge using information extracted from sensor inputs; (b) probabilistically model the uncertainty in sensor input processing and navigation; and (c) use domain knowledge to revise probabilistic beliefs, exploiting positive and negative observations to identify situations in which the assigned task can no longer be pursued. All algorithms are evaluated in simulation and on mobile robots locating target objects in indoor domains.

1 Introduction

Mobile robots are increasingly being deployed to collaborate with humans in domains such as search and rescue, and reconnaissance. These domains characterized by non-determinism and unforeseen changes frequently make it difficult for robots to process all sensor inputs or operate without substantial domain knowledge. At the same time, humans are unlikely to have the time and expertise to provide accurate domain knowledge or elaborate feedback. Widespread deployment of robots in complex real-world domains thus poses fundamental knowledge representation and reasoning (KRR) challenges—robots need to: (a) represent, revise and reason with incomplete knowledge; (b) automatically adapt sensing and navigation to the task at hand; and (c) learn from unreliable, high-level human feedback.

Although there is a rich body of research in KRR, the research community is fragmented. For instance, declarative programming paradigms provide commonsense reasoning capabilities, but do not support probabilistic modeling of uncertainty, which is critical in many robot application domains. In parallel, algorithms based on probabilistic graphical models are being designed to quantitatively model the uncertainty in sensing and acting on robots, but it is difficult to use these algorithms for commonsense reasoning. Furthermore, algorithms for combining logical and probabilistic reasoning do not provide the desired expressiveness for commonsense reasoning capabilities such as non-monotonic reasoning and default reasoning. Our prior work described an architecture that combined the knowledge representation and inference capabilities of *Answer Set Programming* (ASP), a declarative programming paradigm, with the probabilistic decision-making capabilities of *partially observable Markov decision processes* (POMDPs) (Zhang, Sridharan, and Bao 2012). This paper extends the architecture by making the following contributions:

- Richer representation of incomplete domain knowledge, incrementally revising the knowledge base (KB) using information extracted from sensor inputs.
- Principled generation of prior beliefs from the KB, merging these beliefs with POMDP beliefs to adapt sensor input processing and navigation to the task at hand.
- Modeling and learning from positive and negative observations, identifying situations in which the current task should no longer be pursued.

All algorithms are evaluated in simulation and on physical robots visually localizing target objects in indoor domains.

2 Related Work

Algorithms based on probabilistic graphical models such as POMDPs have been used to model the uncertainty in real-world sensing and navigation, enabling the use of robots in offices and hospitals (Göbelbecker, Gretton, and Dearden 2011; Pineau et al. 2003). For computational efficiency, researchers have developed algorithms that decompose complex problems into a hierarchy of (tractable) simpler problems (Pineau et al. 2003; Zhang, Sridharan, and Washington 2013). However, it is still challenging to use probabilistic graphical models in large, complex state-action spaces, and these algorithms do not readily support representation of, and reasoning with, commonsense knowledge.

Research in declarative programming paradigms such as ASP has provided appealing capabilities for non-monotonic logical reasoning and default reasoning (Baral 2003; Gelfond 2008). Although these algorithms have been used in robotics (Chen et al. 2010; Erdem, Aker, and Patoglu 2012), they do not support probabilistic modeling of the consider-

able uncertainty in real-world sensing and navigation. Principled algorithms have been developed to combine logical and probabilistic reasoning, e.g., probabilistic first-order logic (Halpern 2003), Markov logic networks (Richardson and Domingos 2006), and a probabilistic extension to ASP (Baral, Gelfond, and Rushton 2009). Robotics researchers have also developed algorithms that use commonsense knowledge and semantic maps (Hanheide et al. 2011), or support logical and probabilistic reasoning to control a robot's behavior (Göbelbecker, Gretton, and Dearden 2011). However, these algorithms are limited in their ability to support the desired KRR capabilities. Algorithms based on first-order logic do not provide the desired expressiveness for capabilities such as default reasoning, e.g., it is not always necessary, or even possible to express all forms of uncertainty and degrees of belief quantitatively. Other algorithms based on logic programming do not support all desired capabilities such as: causal reasoning; incremental addition of probabilistic information; reasoning with large probabilistic components; or dynamic addition of random variables with different ranges. As a step towards addressing these challenges, our prior work demonstrated the integration of ASP with POMDPs on mobile robots (Zhang, Sridharan, and Bao 2012), but did not fully support incremental knowledge revision and Bayesian belief merging. The architecture described in this paper addresses these limitations.

3 Proposed Architecture

Figure 1 shows our control architecture. The ASP knowledge base (KB) represents domain knowledge (Section 3.1), while POMDPs probabilistically model a subset of state space relevant to the task at hand (Section 3.2). Logical inference in ASP is used to: (a) compute and model prior beliefs relevant to the task at hand as a *Dirichlet* distribution; and (b) identify eventualities not considered by the POMDP formulation based on learned *Beta* distributions. The Dirichlet distribution supports Bayesian merging with POMDP beliefs (Section 3.3), while Beta distributions help robots use both positive and negative observations, terminating tasks early if they can no longer be accomplished (Section 3.4).

The robot uses a learned *policy* to map the merged beliefs to action choices. Action execution causes the robot to move and process images to obtain observations. Observations are also obtained from passive sensors (e.g., range finders) and from high-level human feedback. Observations made with high certainty are (proportionately) more likely to update the KB, while other observations update POMDP beliefs. Human feedback is solicited based on availability and need (computed using entropy of POMDP beliefs). This architecture is illustrated below in the context of robots localizing (i.e., computing the locations of) objects in indoor domains, but the integration of knowledge representation, non-monotonic logical inference and probabilistic decision-making is applicable to many domains.

3.1 Logical reasoning with ASP

Answer Set Programming is a declarative programming paradigm that can represent recursive definitions, causal relations, and other constructs that occur frequently in non-mathematical domains, and are difficult to express in classical logic (Baral 2003; Gelfond 2008). ASP's signature is a tuple of sets: $\Sigma = \langle \mathscr{O}, \mathscr{F}, \mathscr{P}, \mathscr{V} \rangle$ that define names of objects, functions, predicates and variables available for use. The signature is augmented by *sorts* (i.e., *types*) such as: `room/1`, `object/1`, `class/1`, and `step/1` (for temporal reasoning). An ASP program is a collection of statements describing objects and relations between them. An *answer set* is a set of ground literals that represent beliefs of an agent associated with the program. ASP support *default reasoning*, i.e., conclusions can be drawn due to lack of evidence to the contrary, by introducing new connectives: *default negation* (negation by failure) and *epistemic disjunction*, e.g., unlike "`¬ a`" (classical negation) that implies "*a is believed to be false*", "`not a`" only implies "*a is not believed to be true*"; and unlike "`p ∨ ¬p`" in propositional logic, "`p or ¬p`" is not a tautology.

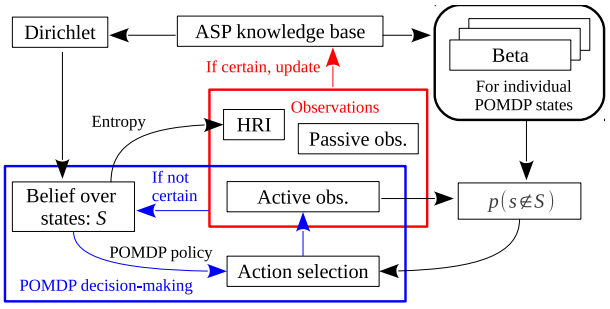


Figure 1: Overview of proposed control architecture.

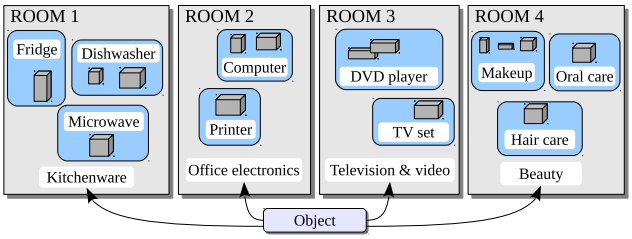


Figure 2: Pictorial representation of a hierarchy of objects.

For indoor target localization, the robot learns a domain map and semantic labels for rooms (e.g., "kitchen", "office"). The hierarchy of objects is learned incrementally from sensor inputs, online repositories and human feedback. Figure 2 illustrates a hierarchy of object classes, which is revised using observations—leaf nodes (specific object instances) can be added or removed and classes (leaf nodes' parents are *primary classes*) can be merged.

The robot models and reasons about aspects of the domain that do not change (*statics*) and can change (*fluents*), using *predicates* defined in terms of the sorts of their arguments: (1) `is(object, class)` describes class membership of a specific object, e.g., `is(printer1, printer)`; (2) `subclass(class, class)` denotes class hierarchy; (3) `in(object, room)` describes the room location of an object; (4) `exists(class, room)`, implies that an object of a class exists in a room; and (5) `holds(fluent, step)` implies a fluent is true at a particular timestep. Predicates are applied recursively when appropriate. The KB also includes

(currently hand-coded) *reasoning* and *inertial* rules such as:

(1) holds(exists(C,R),I) ← holds(in(O,R),I), is(O,C).
(2) holds(exists(C1,R),I) ← holds(exists(C2,R),I), subclass(C2,C1).
(3) ¬holds(in(O,R2),I) ← holds(in(O,R1),I), R1!=R2.
(4) holds(in(O,R1),I+1) ← holds(in(O,R1),I), not holds(in(O,R2),I+1), R1!=R2.

The first rule implies that if 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 first rule in the object hierarchy. The third rule implies that object locations are unique, while the last rule implies that if object O is in room R1, it will continue to be there unless it is known to be elsewhere.

Consider the following example of non-monotonic reasoning in ASP:

step(1..end).
is(printer1,printer).
holds(in(printer1,lab),1).

Reasoning in ASP produces the following answer set (existing facts not listed): `holds(in(printer1, lab), 2). holds(exists(printer, lab), 2).` Now, consider adding a statement: `holds(in(printer1, office), 2)` about printer1's location in the second time step. Reasoning in ASP produces a new answer set (existing facts not repeated); we observe that the outcome of the previous inference step has been revised:

¬holds(in(printer1,lab),2).
holds(exists(printer,office),2).

Next, consider modeling the default: "books are normally in the library" with cookbooks being a *weak exception*:

in(X,library) ← book(X), not ab(d_{in}(X)). % Default
book(X) ← textbook(X). book(X) ← cookbook(X).
ab(d_{in}(X)) ← cookbook(X).
textbook(prml). cookbook(spices). % Specific data

where ab(d(X)) implies "X is abnormal w.r.t d". The result of inference in this example is: `in(prml, library)`, and *no claims are made about* `spices`, i.e., it is unknown if cookbook `spices` exists in the library or not. These capabilities are rather useful for real-world human-robot collaboration. Furthermore, if incorrect information is added to the KB, inconsistencies are identified and corrected by subsequent sensor inputs or by posing queries.

3.2 Uncertainty modeling with POMDPs

To localize objects, robots have to move and analyze images of different scenes. The uncertainty in sensor input processing and navigation is modeled using hierarchical POMDPs in Figure 3. For a given target, the 3D area (e.g., multiple rooms) is modeled as a discrete 2D *occupancy grid*, each grid storing the probability of target existence. The visual sensing (VS)-POMDP plans an action sequence that maximizes information gain by analyzing a sequence of scenes. Executing an action in the VS policy causes the robot to move and analyze a specific scene. For any scene, the (SP)-POMDP with one or more levels is created and solved automatically to apply a sequence of processing algorithms on a sequence of regions of images of the scene. The SP policy's terminal action causes a VS belief update and an action selection. This formulation uses our prior work that introduced *convolutional policies* and learned observation functions for *automatic belief propagation* between levels of the hierarchy and *real-time model creation* in each level (Zhang, Sridharan, and Washington 2013). As a result, robots automatically, reliably and efficiently adapt visual processing and navigation to the task at hand.

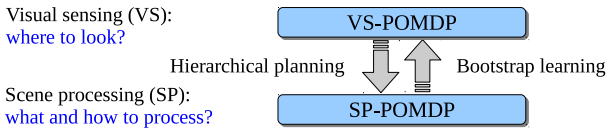


Figure 3: Hierarchical POMDPs for visual processing.

3.3 Generating Prior Beliefs from Answer Sets

This section describes the generation of prior beliefs from the ASP KB. For visually localizing objects in a set of rooms, prior belief of existence of these objects in rooms is based on: (a) knowledge of the hierarchy of object classes and specific domain objects; and (b) postulates that capture object co-occurrence relationships. Some postulates (and their representation) may need to be revised when using this approach in other domains.

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

$$perception = ln(stimulus) + const$$

This law is applicable to visual processing, the primary source of information in this paper. For a target object, support for its existence in a room is thus given by:

$$\psi_n = \begin{cases} 0 & if \;\; a_n = 0 \\ ln(a_n) + \xi & otherwise \end{cases} \quad (1)$$

where a_n is the number of (known) objects of the primary class (of the target) in the room, and $\xi = 1$ corresponds to *const* above. Certain domain objects may be exclusive, e.g., there is typically one fridge in a kitchen. Such properties can be modeled by rules in the KB and by including other postulates (e.g., see below).

Postulate 2: As the number of known subclasses of a class increases, the influence exerted by the subclasses on each other decreases proportionately. 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 1 is modified as:

$$\psi_n = \begin{cases} 0 & if \;\; a_n = 0 \\ \dfrac{ln(a_n) + \xi}{\prod_{h=1}^{H_n} W_h} & otherwise \end{cases} \quad (2)$$

where H_n is the height of LCA of candidate primary class and target. For a class node on the path from primary class to the LCA, W_h is the number of siblings at height h; $W_1 = 1$.

Postulate 3: Prior knowledge of existence of objects in primary classes *independently* provides support for the existence of a specific object (in a room). This postulate works well in practice. Prior belief of existence of a target in room k thus sums the evidence from N primary classes:

$$\alpha_k = \sum_{n=1}^{N} \psi_{n,k} \tag{3}$$

The prior belief of existence of the target in the set of rooms is modeled as a Dirichlet distribution with parameters α, where α_k is the cardinality of the set of relevant answer set statements obtained through ASP-based inference. The probability density function (pdf) of this K-dimensional Dirichlet distribution is:

$$\mathscr{D}(\mu|\alpha) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_K)} \prod_{k=1}^{K} \mu_k^{\alpha_k - 1} \tag{4}$$

where the Γ (Gamma) function is used for normalization; $\mu_k \in [0,1]$ is a distribution of target's existence in the rooms; and $\alpha_0 = \sum_{k=1}^{K} \alpha_k$. The expectation of this Dirichlet distribution is the prior belief of the target's existence in each room. If event E_k represents the target's existence in room k, the probability of this event based on the Dirichlet prior is:

$$p(E_k|\mathscr{D}) = \mathbb{E}(\mu_k) = \alpha_k/\alpha_0 \tag{5}$$

The Dirichlet distribution models the *conditional* probability of target's existence in each room *given* it exists in the domain. To support learning from positive and negative observations, and reasoning about the target's non-existence in a room or domain, α also initialize Beta distributions that model the existence of the target in each room:

$$\mathscr{B}(\phi_k|\alpha_k,\beta_k) = \frac{\Gamma(\alpha_k+\beta_k)}{\Gamma(\alpha_k)\Gamma(\beta_k)} \phi^{\alpha_k - 1}(1-\phi^{\beta_k - 1}) \tag{6}$$

where β_k is the support for target's non-existence in room k. The expectation of Beta distribution for room k serves as prior belief that the target exists in room k. If event E represents the target's existence in the domain:

$$p(E_k|\mathscr{B}) = \mathbb{E}(\phi_k) = \frac{\alpha_k}{\alpha_k+\beta_k} \tag{7}$$

The probability that the target does not exist in the domain can be derived, assuming E_i and E_j are independent $\forall i \neq j$:

$$p(\neg E) = \prod_k p(\neg E_k|\mathscr{B}) = \prod_k (1 - p(E_k|\mathscr{B})) \tag{8}$$

3.4 Using Positive and Negative Observations

Inference based on lack of evidence is a significant challenge, but negative observations can help identify eventualities not modeled by the POMDPs. For instance, not observing a target or related objects in multiple rooms can be used to reason about the target's absence in the domain; this is not computed in the standard POMDP models and introducing a (special) terminal state invalidates invariance properties exploited for computational efficiency.

Let D be the event that the target is detected; and FoV the event that target is in the robot's field of view. The objective is to calculate $p(\neg E|D)$ and $p(\neg E|\neg D)$, and thus $p(E|D)$ and $p(E|\neg D)$, given the POMDP belief state B and action a. To do so, the distribution of target existence and non-existence (in the domain) are updated along with the POMDP belief update. The prior beliefs computed in Section 3.3 are (re)interpreted as beliefs conditioned on the existence (or non-existence) of the target in the domain.

Negative Observations: Not detecting a target at a specific location should not change the probability of target's existence outside the robot's field of view, which is the product of the probability that the target exists in the domain, and the probability that the target exists outside the robot's view given its existence in the domain—the latter value can be computed from POMDP beliefs. The reduction in the probability of target's existence in the field of view will be added to the probability of target's non-existence in the domain:

$$\begin{aligned} p(\neg E|\neg D) &= p(\neg E) + p(E)(p(FoV|E) - p'(FoV|E)) \\ &= p(\neg E) + p(E)(\sum_{s_i \in \Lambda(a)} B(s_i) - \sum_{s_i \in \Lambda(a)} B'(s_i)) \end{aligned} \tag{9}$$

where $B(s_i)$ and $B'(s_i)$ are POMDP beliefs of state s_i before and after the update using this observation; Λ is the set of states that imply that the target is in the robot's field of view.

Positive Observations: A positive observation should increase POMDP beliefs in the field of view, and decrease: (a) beliefs outside this region; and (b) probability of target's non-existence in domain. The posterior probability of target's non-existence is computed using probabilities of the target being detected given that it exists (or does not exist) in the domain. The probability of the target being detected when it does not exist is a fixed probability of false-positive observations. The probability of target being detected when it exists depends on whether it is inside or outside the robot's field of view. If target is in the field of view, probability of a positive observation is given by the POMDP observation functions; if it is outside the field of view, this is the probability of false-positives:

$$p(\neg E|D) = \frac{p(D|\neg E)p(\neg E)}{p(D|E)p(E) + p(D|\neg E)p(\neg E)} \tag{10}$$

The conditional probability that target is detected given that it exists (does not exist) is:

$$\begin{aligned} p(D|E) &= p(D|E,FoV)p(FoV|E) \\ &\quad + p(D|E,\neg FoV)p(\neg FoV|E) \\ &= p(D|FoV)p(FoV|E) + p(D|\neg FoV)p(\neg FoV|E) \\ &= \sum_{s_i \in \Lambda(a)} p(D|s_i,a)B(s_i) + \varepsilon \sum_{s_i \notin \Lambda(a)} B(s_i) \\ P(D|\neg E) &= P(D|\neg E,FoV)P(FoV|\neg E) \\ &\quad + P(D|\neg E,\neg FoV)P(\neg FoV|\neg E) \\ &= P(D|\neg E,\neg FoV) = \varepsilon \sum_{s_i \notin \Lambda(a)} B(s_i) \end{aligned} \tag{11}$$

where ε is the probability of detecting a target when it does not exist; $p(D|s_i,a)$ is from POMDP observation functions.

3.5 Belief Merging

The KB contains domain knowledge, including information not directly relevant to the current task, while the POMDP belief summarizes all observations directly related to the current task. Our prior work heuristically generated a belief distribution from answer sets and (weighted) averaged it with POMDP beliefs (Zhang, Sridharan, and Bao 2012). In this paper, the use of Dirichlet distribution to extract prior beliefs from answer sets supports Bayesian merging:

$$p'(E_k) = \frac{p(E_k|\mathscr{D}) \cdot p(E_k)}{\sum_i p(E_i|\mathscr{D}) \cdot p(E_i)} \tag{12}$$

where $p(E_k)$ is the probability that target is in room k based on POMDP beliefs.

4 Experimental Results

Experiments were designed to evaluate two hypotheses: (H1) combining ASP and POMDP improves target localization (accuracy and time) in comparison with the individual algorithms; and (H2) using positive and negative observations supports early termination of trials that should no longer be pursued. While evaluating H1, models of positive and negative observations are not included. When ASP and POMDPs are used, Bayesian belief merging is compared with: (a) not using the prior beliefs from ASP KB; (b) using relative trust factors to merge beliefs (Zhang, Sridharan, and Bao 2012); and (c) using weights drawn from the Dirichlet distribution to merge beliefs.

Since it is a challenge to run many trials on robots, the architecture was also evaluated extensively in simulation, using learned object models and observation models to realistically simulate motion and perception. For instance, a simulated office domain consisted of four rooms connected by a surrounding hallway in a 15×15 grid. Fifty objects in 10 primary classes (of office electronics) were simulated, and some objects were randomly selected in each trial as targets whose positions were unknown to the robot. The robot revises the KB, including the basic object hierarchy mined from repositories, during experimental trials. Each data point in the results described below is the average of 5000 simulated trials. In each trial, the robot's location, target objects and their locations are chosen randomly. A trial ends when belief in a grid cell exceeds a threshold (e.g., 0.80). Some trials include a time limit, as described below.

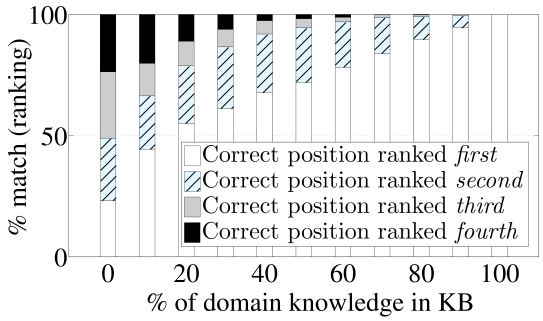
To evaluate hypothesis H1, we measured the target localization accuracy and time when the robot used: (a) only ASP; (b) only POMDPs; and (c) ASP and POMDPs. First, the robot used domain knowledge and Equations 1-4 to infer the target objects' locations. ASP can only infer object existence in rooms and cannot infer specific locations of objects in rooms. Figure 4(a) shows that when the robot has all the domain knowledge (except the target), i.e., 100 along x-axis, it can correctly infer the room containing the object. The accuracy decreases when the domain knowledge decreases, e.g., with 50% domain knowledge, the robot can correctly identify the target's (room) location with 0.7 accuracy. However, even with 50% domain knowledge, the correct room location is in the top two choices in 95% of the trials. One key benefit of using ASP-based inference is thus the significant reduction in target localization time.

Once ASP identifies likely locations of target objects, POMDPs focus the robot's sensing and navigation to identify specific locations of objects. Figure 4(b) summarizes these experiments as a function of the amount of domain knowledge included and used to generate prior beliefs—the specific domain knowledge to be added is selected randomly in the simulation trials. These trials have a time limit of 100 units. Trials corresponding to 0 on the x-axis represent the use of only POMDPs. Combining prior beliefs extracted from answer sets with POMDP beliefs significantly increases target localization accuracy—the robot uses prior knowledge to recover from incorrect observations, and uses observations to revise existing knowledge. As the robot acquires and uses more knowledge, localization accuracy steadily improves—with all relevant domain knowledge (except the target), accuracy is 0.96, and errors correspond to trials with objects at the edge of cells.

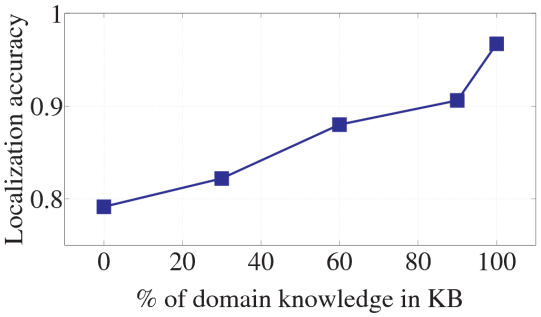
Next, to compare the Bayesian approach of generating and merging beliefs with other strategies, the KB is initialized with 20% domain knowledge. Periodically, information about some randomly chosen objects is added to simulate learning. Inference in ASP produces new answer sets that provide prior beliefs to be merged with POMDP beliefs to guide subsequent actions. Results are summarized in Figure 5(a)—x-axis represents the localization error in units of grid cells in the simulated domain, while y-axis represents % of trials with errors below a value. For instance, with our approach, more than 80% of the trials report an error of ≤ 5 units. These results indicate that our belief generation and merging strategy results in much lower localization errors than: not using ASP, our previous approach that used "trust factors" to merge beliefs, and the Dirichlet-based weighting scheme that assigns weights to ASP and POMDP beliefs based on the degree of correspondence with the Dirichlet distribution. Similar results were observed with different levels of (initial) knowledge—assigning undue importance to ASP or POMDP beliefs can hurt performance even when significant domain knowledge is available.

Experiments were then conducted to evaluate hypothesis H2. The KB is fixed, and in each target localization trial, the target is randomly selected to be present or absent. A baseline policy (for comparison) is designed to use ASP and POMDP beliefs (similar to experiments described above); this policy claims absence of target if it cannot be found within the time limit. Figure 5(b) shows that exploiting positive and negative observations results in much higher target localization accuracy while significantly lowering localization time. If positive and negative observations are not exploited, comparable localization accuracy is only obtained by allowing a much larger amount of time for localization.

We also evaluated the architecture on a physical (wheeled) robot deployed in an office domain with multiple offices, corridors, labs and kitchen. The robot learns the domain map and localizes target object(s) by processing sensor inputs. Videos are available online:
http://youtu.be/EvY_Jt-5BqM
http://youtu.be/DqsR2qDayGQ

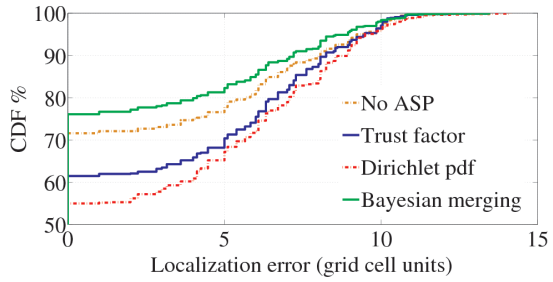


(a) ASP-based target localization.

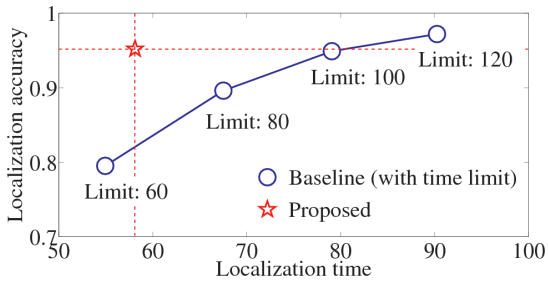


(b) ASP+POMDP target localization.

Figure 4: (a) Target localization accuracy using only ASP; (b) Target localization accuracy as a function of % of domain knowledge in the KB. Combining ASP and POMDP significantly increases accuracy in comparison with just using POMDPs.



(a) Belief merging strategies.



(b) Using positive and negative observations.

Figure 5: (a) Using Dirichlet distribution for extracting prior beliefs from answer sets, and Bayesian belief merging result in lowest localization errors; (b) Using positive and negative observations results in higher accuracy and lower localization time.

5 Conclusions

This paper described an architecture that integrates the knowledge representation and reasoning capabilities of ASP with the probabilistic uncertainty modeling capabilities of hierarchical POMDPs. Experimental results in the context of a mobile robot localizing target objects in indoor domains indicates that the architecture enables robots to: (a) represent, revise and reason with incomplete domain knowledge acquired from sensor inputs and human feedback; (b) generate and merge ASP-based beliefs with POMDP beliefs to tailor sensing and navigation to the task at hand; and (c) fully utilize all observations. Future work will further explore the planning and diagnostics capabilities of ASP, and investigate the tighter coupling of declarative programming and probabilistic reasoning towards reliable and efficient human-robot collaboration in real-world domains.

References

Baral, C.; Gelfond, M.; and Rushton, N. 2009. Probabilistic Reasoning with Answer Sets. *Theory and Practice of Logic Programming* 9(1):57–144.

Baral, C. 2003. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press.

Chen, X.; Ji, J.; Jiang, J.; Jin, G.; Wang, F.; and Xie, J. 2010. Developing High-Level Cognitive Functions for Service Robots. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Erdem, E.; Aker, E.; and Patoglu, V. 2012. Answer Set Programming for Collaborative Housekeeping Robotics: Representation, Reasoning, and Execution. *Intelligent Service Robotics* 5(4):275–291.

Gelfond, M. 2008. Answer Sets. In Frank van Harmelen and Vladimir Lifschitz and Bruce Porter., ed., *Handbook of Knowledge Representation*. Elsevier Science. 285–316.

Göbelbecker, M.; Gretton, C.; and Dearden, R. 2011. A Switching Planner for Combined Task and Observation Planning. In *National Conference on Artificial Intelligence*.

Halpern, J. 2003. *Reasoning about Uncertainty*. MIT Press.

Hanheide, M.; Gretton, C.; Dearden, R.; Hawes, N.; Wyatt, J.; Pronobis, A.; Aydemir, A.; Gobelbecker, M.; and Zender, H. 2011. Exploiting Probabilistic Knowledge under Uncertain Sensing for Efficient Robot Behaviour. In *International Joint Conference on Artificial Intelligence*.

Pineau, J.; Montemerlo, M.; Pollack, M.; Roy, N.; and Thrun, S. 2003. Towards Robotic Assistants in Nursing Homes: Challenges and Results. In *RAS Special Issue on Socially Interactive Robots*, volume 42, 271–281.

Richardson, M., and Domingos, P. 2006. Markov Logic Networks. *Machine learning* 62(1).

Zhang, S.; Sridharan, M.; and Bao, F. S. 2012. ASP+POMDP: Integrating Non-monotonic Logical Reasoning and Probabilistic Planning on Robots. In *International Conference on Development and Learning and Epigenetic Robotics*.

Zhang, S.; Sridharan, M.; and Washington, C. 2013. Active Visual Planning for Mobile Robot Teams using Hierarchical POMDPs. *IEEE Transactions on Robotics* 29(4).