Active Visual Planning for Mobile Robot Teams Using Hierarchical POMDPs

Shiqi Zhang, Mohan Sridharan, and Christian Washington

***Abstract*—Key challenges to widespread deployment of mobile robots include collaboration and the ability to tailor sensing and information processing to the task at hand. Partially observable Markov decision processes (POMDPs), which are an instance of probabilistic sequential decision-making, can be used to address these challenges in domains characterized by partial observability and nondeterministic action outcomes. However, such formulations tend to be computationally intractable for domains that have large complex state spaces and require robots to respond to dynamic changes. This paper presents a hierarchical decomposition of POMDPs that incorporates adaptive observation functions, constrained convolutional policies, and automatic belief propagation, enabling robots to retain capabilities for different tasks, direct sensing to relevant locations, and determine the sequence of sensing and processing algorithms best suited to any given task. A communication layer is added to the POMDP hierarchy for belief sharing and collaboration in a team of robots. All algorithms are evaluated in simulation and on physical robots, localizing target objects in dynamic indoor domains.**

***Index Terms*—Bayesian methods, hierarchical systems, intelligent robots, Markov processes, multirobot systems, planning, robot vision systems.**

I. Introduction

MOBILE robots are increasingly being deployed in real-world application domains such as disaster rescue and medicine due to the ready availability of high-fidelity sensors and the development of sophisticated algorithms to process sensor inputs. Key challenges to widespread deployment of robots include collaboration and the ability to adapt sensing and information processing to the task at hand. In real-world domains characterized by partial observability, nondeterministic action outcomes, and unforeseen dynamic changes, a robot equipped with multiple sensors cannot reliably observe the entire domain from a fixed location. In addition, information can be extracted from sensor inputs using algorithms with varying levels of uncertainty and computational complexity. It is, therefore, a challenge for robots to respond to dynamic changes by fully exploiting information that is relevant to the task at hand. Although humans can provide rich information about the task and domain, humans may not have the time and expertise to provide elaborate and accurate feedback in complex domains. Furthermore, multirobot collaboration poses additional challenges because communication may be unreliable and robots in the team may possess different capabilities.

The long-term objective of our research is to enable robots to collaborate with nonexpert humans, automatically acquiring and using relevant sensor inputs and human feedback based on need and availability. Toward this objective, this paper focuses on reliable and efficient vision-based sensing, information processing, and collaboration in a team of mobile robots. Our prior work introduced a novel two-layered hierarchical decomposition of partially observable Markov decision processes (POMDPs) [15], enabling a robot and a human to jointly reason about and manipulate objects in simplistic tabletop scenarios [29]. This paper describes more recent research corresponding to the following contributions [32].

1) Each robot automatically directs sensing to relevant locations and determines the sequence of information-processing algorithms appropriate for any given task.
2) A communication layer is added to the POMDP hierarchy to enable each robot to probabilistically merge own beliefs with the information communicated by teammates.

The remainder of this paper is organized as follows. Section II discusses related work in visual planning and collaboration, while Section III describes our POMDP hierarchy. Experimental results in simulation and on robots are presented in Section IV, followed by a discussion of conclusions and future work in Section V.

II. Related Work

Classical planning algorithms typically require prior knowledge of domain states, action outcomes, and contingencies [12], and considerable research has been done to relax these constraints [3], [24]. The planning with knowledge and sensing (PKS) planner [24] uses a first-order language to describe actions in terms of their effect on the agent's knowledge, rather than their effect on the world. The system is nondeterministic because the agent's knowledge of state is not uniquely determined by the actions performed. The Continual Planning algorithm [3] interleaves planning, execution, and monitoring and postpones reasoning about uncertain states by asserting that action preconditions will be met when that point is reached during plan execution. Replanning occurs if preconditions are not met during execution or are met earlier. However,

Manuscript received July 29, 2012; revised November 27, 2012; accepted March 3, 2013. Date of publication April 1, 2013; date of current version August 2, 2013. This paper was recommended for publication by Associate Editor E. Marchand and Editor D. Fox upon evaluation of the reviewers' comments. This work was supported in part by the Office of Naval Research Science of Autonomy Award N00014-09-1-0658 and the National Science Foundation Grant CNS-1005212. Any opinions and conclusions reported in this paper are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research or the National Science Foundation.

S. Zhang and M. Sridharan are with the Department of Computer Science, Texas Tech University, Lubbock, TX 79409-3104 USA (e-mail: shiqi.zhang6@gmail.com; mohan.sridharan@ttu.edu).

C. Washington is with Louisiana State University, Baton Rouge, LA 70814 USA (e-mail: cwash24@lsu.edu).

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

Digital Object Identifier 10.1109/TRO.2013.2252252

it is a challenge to use such algorithms to support capabilities that are required in robot application domains, e.g., the ability to accumulate evidence by applying information-processing algorithms more than once.

The ability of POMDPs to model domains characterized by partial observability and nondeterminism has been used to plan actions in tasks such as robot navigation and image interpretation [10], [19]. Probabilistic graphical models have also been used for sensor placement and active sensing [17]. These algorithms typically require substantial knowledge of task and domain and/or require human supervision, but human participants may not have the time and expertise to provide elaborate feedback or accurate domain knowledge. In addition, POMDP formulations of complex domains frequently result in exponentially increasing state spaces, and modern POMDP solvers can (in the worst case) have an exponential time complexity. Researchers have, hence, imposed structure (or hierarchy) on problem domains to support POMDP formulations [10], [25]. Theocharous [31] modeled hierarchical POMDPs as dynamic Bayesian networks and used multiresolution spatial maps for robot navigation. Pineau *et al*. [25] used a POMDP hierarchy for behavior control of a robot assistant, with bottom-up planning and top-down action execution. However, these algorithms still require expert supervision to create the hierarchy and associated models. To make POMDP formulations computationally tractable, researchers have also introduced factored representations that separate the fully and partially observable portions of the state [21]. In parallel, attention is being devoted to knowledge representation and the use of commonsense knowledge in robotics [1], [8], [11]. More recent work is focusing on combining logical and probabilistic reasoning for task and motion planning on robots [13], [16]. However, substantial human supervision is required to acquire and revise domain knowledge, which may be a challenge in complex domains.

Many algorithms have been developed for multirobot and multiagent collaboration [22], [23]. Researchers have also developed algorithms for decentralized information fusion, e.g., the decentralized delayed-state information filter enables heterogeneous agents to fuse information [7], and a decentralized information-gathering algorithm has been able to provide scalability, robustness, and modularity [20]. However, these algorithms are not well suited to model the partial observability of robots deployed in dynamic domains. Although decentralized POMDPs (Dec-POMDPs) are being used for multiagent collaboration [18], [28], the computational complexity of solving Dec-POMDPs is higher than that of default POMDPs [2]. Another option is to use interactive POMDPs (I-POMDPs) [14] that enable agents to model the behavior (and preferences) of other agents as interactive beliefs with arbitrary levels of nesting. However, I-POMDPs have high computational complexity and require significant domain knowledge. An important factor in multirobot collaboration is the unreliability of communication between robots. Research has shown that complex communication strategies do not necessarily benefit robot teams engaged in collaborative tasks [27]. The POMDP hierarchy described in this paper supports automatic belief propagation and model generation, enabling robots to adapt sensing and information processing to the task at hand. The hierarchy is augmented with a communication layer for belief sharing and multirobot collaboration.

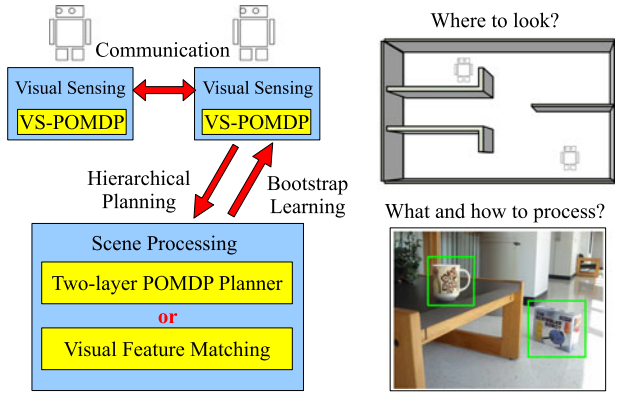

Fig. 1. Overview of the POMDP hierarchy and scenario.

III. Problem Formulation

This section describes the problem domain, formulates the visual planning task using hierarchical POMDPs, and describes the approach for multirobot collaboration.

A. Problem Domain and Hierarchy Overview

The visual planning and collaboration algorithms are illustrated in the context of a team of robots localizing objects in large, complex, and dynamic indoor office domains. Fig. 1 summarizes the POMDP hierarchy for sensing, information processing, and collaboration. Each robot uses a POMDP hierarchy to locate target objects. The top-level visual sensing (*VS*)-*POMDP* computes the sequence of 3-D scenes to process to locate a specific target object (see Section III-B). For any chosen scene, the robot moves to an appropriate location and uses a scene processing (*SP*)-*POMDP* comprising one or two layers (depending on scene complexity) to determine the sequence of information-processing algorithms to apply on a sequence of regions of interest (ROIs) in images of the scene (see Section III-C). Each robot also uses the communication layer to share beliefs and collaborate with teammates (see Section III-D).

B. Visual Sensing Partially Observable Markov Decision Processes for Visual Search

In an office or a home, a robot has to move and analyze different scenes because target objects may exist in different locations in a room or in different rooms. Consider the situation where the robot has learned a domain map based on (laser) range data, using a simultaneous localization and mapping algorithm to revise the map and compute own position in the map. To localize a specific target, the 3-D area is represented as a discrete 2-D *occupancy grid*. Each cell in the grid stores the probability of occurrence of the target object in that cell. The size of each cell is based on the field of view of the sensor (i.e., camera). The VS-POMDP poses sensing as the task of determining a sequence

of actions (i.e., scenes to process) that maximizes information gain or equivalently reduces entropy of the probability distribution over all cells. The VS-POMDP tuple $\langle S, A, T, Z, O, R\rangle$ to localize an object in a domain with N cells is defined as follows.

1) S: $\{s_i,\ i \in [1, N]\}$ is the state vector; s_i corresponds to the event that the target object is in cell i.
2) A: $\{a_i, i \in [1, N]\}$ is the set of actions. Executing a_i causes the robot to move to and analyze cell i.
3) T: $S \times A \times S' \rightarrow [0, 1]$ is the state transition function, i.e., an identity matrix for actions that do not change the state.
4) Z: {present, absent} is the set of observations that indicates the presence or absence of the target in a cell.
5) O: $S \times A \times Z \rightarrow [0, 1]$ is the observation function, which is learned (see below).
6) R: $S \times A \rightarrow \mathbb{R}$ is the reward specification that is based on belief entropy (see below).

Since the robot cannot observe the true state, it maintains a *belief state*: a probability distribution over the underlying set of states. The *entropy* of belief distribution B_t is given by

$$\mathscr{H}(B_t) = -\sum_{i=1}^{N} b_t^i log(b_t^i) \tag{1}$$

where b_t^i is the ith entry of the belief distribution over the learned domain map at time t. The reward of action a_t is defined as the reduction in entropy between belief state B_{t-1} and the resultant belief state B_t after executing action a_t:

$$\begin{aligned} R(a_t) &:= \mathscr{H}(B_{t-1}) - \mathscr{H}(B_t) \\ &= \sum_k b_t^k \log(b_t^k) - \sum_j b_{t-1}^j \log(b_{t-1}^j). \end{aligned} \tag{2}$$

When nothing is known about the target object's location, the belief is uniformly distributed and entropy is maximum. Entropy reduces as the belief distribution converges to states likely to be the target's location. To enable robots to exploit local symmetries in visual processing for computational efficiency, costs associated with robot motion are included in a separate postprocessing step [see (9)].

The observation function models the probability of target detection as a function of the robot position and target position:

if $isBlocked(s_j, a_k)$

$$O(z_i = present, s_j, a_k) = Pr(z_i = present | s_j, a_k) = \beta$$

else

$$O(z_i = present, s_j, a_k) = \eta \cdot exp\left\{-\frac{1}{2}\lambda \alpha^T \Sigma^{-1} \alpha\right\}$$

$$O(z_i = absent, s_j, a_k) = 1 - O(z_i = present, s_j, a_k) \tag{3}$$

where the probability of observing the target 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 whose mean is the target's position. The term α is the offset between target position and the cell being examined, and λ represents the sensitivity of visual recognition to distance. The covariance of the Gaussian, i.e., Σ, represents the uncertainty in the observations, e.g., a higher uncertainty is associated with an observation of the target at a greater distance. The factor η is a normalizer. If there is an obstacle between robot and the target, i.e., $isBlocked(s_j, a_k)$, β is a small probability that the target can still be observed. This observation function is learned from the lower level POMDPs in a semisupervised manner [29]—it is used to perform belief updates based on observations and to generate observations in the simulated experiments. As with the motion costs, the orientation of target observations is included in a postprocessing step [see (16)].

Given the model parameters, belief update in a POMDP is based on Bayesian inference:

$$B_{t+1}(s') = \frac{O(s', a_{t+1}, o_{t+1}) \sum_s T(s, a_{t+1}, s') \cdot B_t(s)}{p(o_{t+1}|a_{t+1}, b_t)}. \tag{4}$$

POMDP solvers use such a model to compute a *policy* π^H: $B_t \mapsto a_{t+1}$ that maps belief states to actions. In the VS-POMDP, an existing implementation of policy gradient algorithms [4] is used to compute the stochastic policy that maximizes entropy reduction over a planning horizon.

1) Convolutional Policy (Kernel Extraction): Practical domains can change and have different shapes and sizes, and the number of cells can be arbitrarily large; solving POMDP formulations of such domains can be computationally expensive. The proposed hierarchy addresses this challenge by learning a convolutional policy kernel that exploits the rotation and shift invariance of visual search. This strategy is motivated by the fact that observations obtained by a robot at any location are primarily influenced by (and modify beliefs about) the neighboring locations [5]. A stochastic policy kernel is, hence, learned from the baseline policy for a small local region:

$$\begin{aligned} \bar{K}(s) &= (\pi^V \otimes\ C_m^K)(s) = \int \pi^V(\tilde{s}) C_m^K(s - \tilde{s}) d\tilde{s} \\ K &= \left(\sum_{\text{states}} \bar{K}\right) \cdot \Big/ W \end{aligned} \tag{5}$$

where π^V is the (baseline) VS policy, C_m^K is a mask of the same size as the kernel being learned, $\bar{K}$ is the unnormalized kernel, W is the count of accumulated weights for each action, and K is the normalized kernel. A small kernel size is chosen to allow generalization to maps of different sizes, and the baseline policy's size is chosen based on computational considerations. No other constraints are imposed on the kernel or map. Consider Fig. 2(a), where a 3×3 policy kernel is extracted from a 5×5 baseline policy: a 2-D matrix whose rows denote actions weights for specific states. Each row is rearranged as a 2-D matrix (of the same size as the map) that stores action weights when focusing on a specific state, decomposing the policy into layers, as shown in the left column of Fig. 2(a). A 3×3 mask C_m^K is convolved with the policy layers, and weights in the region covered by the mask are accumulated, as shown in the middle column of Fig. 2(a). Since weights in cells outside the masked region are not considered, the resultant kernel is normalized (using matrix W) to obtain K, as shown in the right column of Fig. 2(a).

In addition to the learned policy kernel, a small weight value is computed to be assigned (during policy extension) to cells

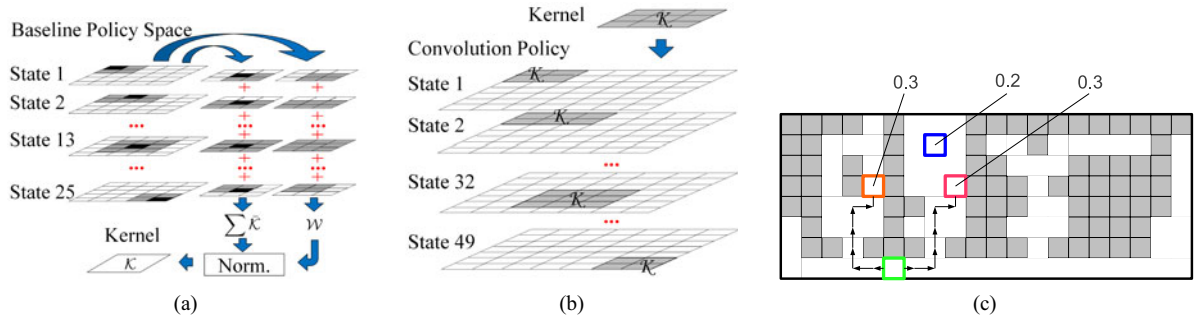

Fig. 2. (a) Extracting 3×3 policy kernel from a 5×5 baseline policy. (b) Using a learned 3×3 policy kernel to generate 7×7 convolutional policy. (c) Hot-spot detection for motion planning.

further away from the center of the mask:

$$W^B = \frac{\sum_{\text{actions}} \sum_{\text{states}} \pi^V - \sum \sum_{\text{states}} \bar{K}}{N_{\text{actions}} \times N_{\text{states}} - sz(W)} \tag{6}$$

where the default weight value is a function of the number of actions N_{actions}, the number of states N_{states}, and the size of the weight matrix $sz(W)$.

When the policy kernel is used to generate policies for larger maps (see below), the number of states covered by the kernel remains unchanged. Since policy weights over the map are normalized, the kernel's effect will be different on maps of different sizes, e.g., it will be much smaller when the size of the map grows larger. A heuristic function is, hence, used to revise the value of W^B such that the ratio of importance assigned to the area covered and left uncovered by the kernel is similar over maps of different sizes:

$$\hat{W}^B = W^B - \ln\left(\frac{N^E_{\text{states}} - sz(W)}{N^B_{\text{states}} - sz(W)}\right) \tag{7}$$

where N^E_{states} and N^B_{states} are the number of states in the large map and baseline kernel, respectively. The natural logarithm function (ln) is used because the conversion of weight values to probabilities is based on a *softmax*-like activation function. Although it may take some time to learn a baseline policy for a small area and extract a policy kernel, this one-time computation does not need to be repeated, unless the properties of the robot's sensors change significantly.

2) Convolutional Policy (Policy Extension): The learned policy kernel is used to compute the policy for larger maps using an efficient convolution operation:

$$\pi^V_C(s) = (K \otimes C^E_m)(s) = \int K(\tilde{s}) C^E_m(s - \tilde{s}) d\tilde{s} \tag{8}$$

where K is the policy kernel, C^E_m is the mask of the same size as the target map, and π^V_C is the convolutional policy. Consider Fig. 2(b), where a learned 3×3 kernel is convolved with a 7×7 mask to generate the policy for a 7×7 map. This policy is generated one layer at a time, by centering the kernel on the state represented by the layer, e.g., there are 49 layers for the 7×7 map. Since the kernel covers (at most) nine cells, other cells are assigned the weight computed in (7), and the policy is normalized. Robots can thus use the policy kernel to generate policies for maps of larger areas in real time.

Mobile robots have to physically move between cells in the map to search for target objects. Sensing, information processing, and actuation on robots are nondeterministic. In addition, robot motion takes time and expends energy. A cost is, hence, assigned to movement by revising each action's (policy) weights during policy execution based on the distance to be traveled and the robot's speed:

$$\hat{w}(a_j) = f(w, d_{A^*}) = w(a_j) \frac{1}{1 + \frac{d_{A^*}(a_i, a_j)}{\text{speed}}} \tag{9}$$

where $w(a_j)$ is the policy weight for action a_j, and $d_{A^*}(a_i, a_j)$ is the distance between the current cell and candidate cell. The A^* algorithm is used to compute the shortest path from current cell to a candidate cell. The modified policy trades off likelihood of localizing a target against the cost of moving to that location. A robot thus does not choose to travel a long distance between two sensing actions, unless it expects to obtain a significant amount of information about the target's location when it reaches the candidate cell. When the domain map changes (e.g., doors are closed or obstacles are moved), the robot also uses this policy reweighting to quickly recompute the distances between cells and revise the action weights before making subsequent action choices. Appropriate values of *speed* can be used in (9) by each robot.

While the revision of action weights captures motion-based costs, hill-climbing is used to make the search more efficient in large maps. Fig. 2(c) shows a domain map discretized into cells, with the green cell being the position of a robot after executing the most recent action. There are three cells in the map with significantly higher weights than other cells: The orange and pink cells have $w = 0.3$ (not $\hat{w}$), and the blue cell has $w = 0.2$. Since the robot's current position is equidistant from the pink and orange cells, these cells have an equal chance of being the next cell visited by the robot. However, it makes sense to visit the pink cell first because it is also close to the blue cell, which is a candidate cell of similar relevance. We, therefore, enable robots to consider the entire path of candidate cells, i.e., the path with the largest summation of $\hat{w}$ values instead of just looking for a target cell with the largest $\hat{w}$. It is, however, infeasible to evaluate all possible paths through all cells in a large map. Our

approach, therefore, detects "hot-spots," i.e., cells with beliefs substantially larger than their immediate neighborhood, and evaluates paths through them. The number of hot-spots (N^{hs}) is a small fraction of the number of cells in the domain map. To compute hot-spots, N^{hs} seeds are selected randomly or initialized based on prior knowledge and refined by hill-climbing to arrive at local maxima such as the orange, blue, and pink cells in Fig. 2(c). The robot considers these hot-spots for further analysis by evaluating paths through them:

$$w^{\mathrm{path}}([h_0, h_1, \ldots, h_{N^{\mathrm{hs}}}]) = \sum_{i=1}^{N^{\mathrm{hs}}} f\left(w(h_i), \sum_{j=1}^{i} d_{A^*}(h_{j-1}, h_j)\right) \tag{10}$$

where h_i is the ith hot-spot, h_0 is the robot's current position, and $w(h_i)$ is the (action) weight of the cell corresponding to hot-spot h_i. The function f is defined in (9). For a set of hot-spots, the robot evaluates all paths through these hot-spots and chooses a cell for analysis, e.g., values of *pink–blue–orange* and *orange–pink–blue* paths in Fig. 2(c) are 0.0672 and 0.0591, respectively, and the pink cell is analyzed next. This path planning does *not* imply that the robot will move through the entire path; the observations made by a robot in a cell revise the belief distribution and planned path. The path planning ensures that the robot's attention is directed toward the most interesting cells.

C. Scene Processing Partially Observable Markov Decision Processes for Scene Processing

Invoking the policy obtained by solving the VS-POMDP for a specific target causes the robot to move to a cell and analyze the corresponding 3-D scene by extracting salient ROIs in an image of the scene. There are two options for scene processing based on scene complexity, i.e., based on number of ROIs in the image and features used to represent objects. In uncluttered scenes with a small number of ROIs, the SP-POMDP has two layers. Each ROI is modeled as a lower level (*LL*)-*POMDP*, where actions are information-processing operators (e.g., detect color). The LL policy provides the sequence of operators to apply on a specific ROI to detect the desired object. The LL policies of all image ROIs are used to automatically create and solve a high-level (*HL*)-*POMDP*, where actions direct the robot's attention to specific ROIs. Executing the HL policy causes the robot to analyze a specific ROI using the corresponding LL policy. Executing the LL policy till termination provides an observation that causes an HL belief update and action choice until presence or absence of the target is established in the image. The creation of POMDP models for these two layers (i.e., HL and LL) has been described in detail in our prior work, which focused on simplistic tabletop scenarios and considered scenes with partially occluded objects [29]. In cluttered scenes with many ROIs, on the other hand, the robot may need to learn sophisticated object models. Scene processing is then formulated as a POMDP that plans the sequence of operators to apply on the image to establish presence or absence of the desired object. Section IV-B provides some examples.

The overall operation of the POMDP hierarchy is as follows: The robot uses the learned domain map to generate the VS-POMDP, which is solved to obtain the VS policy that is used to choose a cell in the domain map for analysis. The robot moves to this cell and processes images of the corresponding 3-D scene using the SP-POMDP with one or two layers, creating and solving the associated POMDP models at run-time. Executing the SP policies until termination provides an observation in the VS-POMDP regarding presence or absence of target in the image. After the belief update, the robot invokes the VS policy to choose a cell (and thus a 3-D scene) for subsequent analysis. This process continues until the object is found or the belief distribution does not converge over a period of time. The key advantage is that automatic belief propagation and model generation in all levels of the hierarchy results in autonomous, reliable, and efficient visual sensing and information processing in complex domains.

D. Multirobot Collaboration

Next, consider a team of robots tasked with localizing one or more target objects. Each robot maintains a separate belief vector for each target. Each robot also uses hierarchical POMDPs (as described previously) to adapt visual sensing and information processing to the task and domain. This section describes a probabilistic approach for the robots to share beliefs and collaborate to locate the desired target objects. The data structure maintained by each robot consists of

$$\{B_i, f_i\},\ \forall i \in [1, |TL|] \tag{11}$$

where B_i is the belief vector for a specific target i among the list of target objects (TL), and f_i is a binary flag that states if the target has been discovered. In addition, each robot stores an action map $\mathscr{M}$, a vector of the same size as the belief vector. Each entry in this vector stores the number of times the robot has visited the corresponding cell in the domain map:

$$\mathscr{M} = \langle m_1, \ldots, m_N \rangle \tag{12}$$

where m_i is the count of the number of times cell i has been visited. The entries in the action map corresponding to locations that have not been visited in the recent past decay over time. As a robot moves to detect a specific target, it updates its action map and uses each observation to update the appropriate belief vector. After such a belief update, the robot communicates with teammates by broadcasting a package that includes current belief vectors for all objects ($\forall i\ B_i$), discovery flags ($\forall i\ f_i$), action map ($\mathscr{M}$), and own position. If the bandwidth is limited, robots can communicate just the changes in the data structure at a low frequency.

A robot cannot completely trust information received from teammates. At the same time, the communicated estimates provide useful information about (possibly large) regions of the domain that the robot has not visited and hence has no knowledge about. In the proposed belief merging scheme, each robot, therefore, assigns probabilistic weights to own beliefs and beliefs communicated by each teammate. The objective is to assign greater importance to estimates communicated by a robot if the robot has recently visited the corresponding region of the map. Each robot, therefore, uses the action map entries as a

probabilistic weight distribution to merge own beliefs with communicated beliefs:

$$b_i^{j,\text{own}} = \frac{m_i^{j,\text{own}} \cdot b_i^{j,\text{own}} + m_i^{j,\text{comm}} \cdot b_i^{j,\text{comm}}}{m_i^{j,\text{own}} + m_i^{j,\text{comm}}}$$
$$\forall j \in [1, N], \quad \forall i \in [1, |TL|] \tag{13}$$

where b_i^j is jth entry of the belief vector corresponding to target i, while $m_i^{j,\text{own}}$ and $m_i^{j,\text{comm}}$ are entries of action maps of the robot and the teammate whose communicated belief is being merged. The action map entries are not merged to prevent rumor propagation among teammates. When one or more robots revise their domain maps in response to changes, data association can be achieved by grounding (i.e., matching) the communicated belief vectors using the corresponding communicated robot locations. Each robot is thus able to assimilate communicated estimates that may complement or contradict own beliefs, and the merged beliefs are revised as each robot's beliefs change. Although this belief merging strategy can (in theory) be sensitive to the order in which the communicated beliefs are merged, it works well in practice.

Each robot also updates the vector of flags representing the discovery of targets (f_i) based on efforts of all teammates:

$$\mathscr{F} = \{f_i^{\text{own}} \| f_i^{\text{comm}}; \forall i \in [1, |TL|]\} \tag{14}$$

where each target is assumed to be found when at least one robot in the team has communicated its discovery (belief in a cell above a threshold) to teammates. There may, hence, be times when a target object is being searched for by more than one team member. This overlap of targets among robots in the team is allowed (intentionally) to ensure effective coverage of target objects by robots in the team. However, in practice, multiple robots rarely search for the same object.

Once a target is discovered, a new target is chosen from the list of undiscovered objects in TL:

$$targetID = \underset{i}{\text{argmax}}\{\max_j B_i(j)\} \tag{15}$$

where the robot identifies target i whose location it is most certain about based on the merged beliefs of all teammates. The robot thus selects the target object that it is likely to locate with the least effort. This choice of a new target can also include a heuristic cost based on distance of travel and relative priority of the remaining targets (if such information is available). These costs can be incorporated as weights on the belief vectors, similar to the policy reweighting performed by (9). The key outcome is that robots in a team are able to reliably and efficiently coordinate their efforts despite unreliable communication. Changes in team composition are addressed automatically, and the lack of communication causes each robot to smoothly transition to operating as if it were the only robot in the team. The next section evaluates the visual sensing, processing, and multirobot collaboration capabilities.

IV. Experimental Setup and Results

This section describes the initial setup and results of experiments in simulation and on physical robots (see Fig. 5). The objective is to evaluate each robot's ability to 1) use the POMDP hierarchy to adapt visual sensing and processing to the task at hand and 2) probabilistically merge own beliefs with communicated beliefs for collaboration. These capabilities are evaluated in the context of robots localizing target objects in indoor domains. Given the prior use of SP-POMDP in other domains [29], this section considers the result of using SP-POMDP (to process an image) as a response for executing an HL action. The VS-POMDP models are solved using policy gradient algorithms in the LibPG library [4]. Since it is not feasible to execute a large number of trials on mobile robots, experiments also included realistic simulations of grid maps of different sizes and teams with different numbers of robots.

Evaluation on robots requires the following initial setup and revisions. First, in an initial semisupervised learning phase, robots apply different information-processing algorithms on images of objects with known labels to learn object models and some POMDP model parameters [29]; some examples of object models are described in Section IV-B. Second, during policy execution, robots compute the relative distance and bearing to objects. Since including orientation as a parameter in the observation set will destroy the local invariance in policy space, the belief update [see (4)] is modified:

$$\textbf{if } \neg target$$
$$B(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{Pr(o|a, b)} = \frac{O(s', a, o) b(s)}{Pr(o|a, b)}$$
$$\textbf{else}$$
$$B(s') = \frac{O(s', \hat{a}, o) \sum_{s \in S} T(s, \hat{a}, s') b(s)}{Pr(o|\hat{a}, b)} = \frac{O(s', \hat{a}, o) b(s)}{Pr(o|\hat{a}, b)} \tag{16}$$

where $B(s')$ is the updated belief for state s' after action a. The belief update can be simplified (as shown) because the transition functions are identity matrices. When a target is detected, the relative distance and bearing are used to find global location of the target in the grid map (based on robot's estimate of own location). The belief update is performed as if the action corresponding to this global location ($\hat{a}$) had been executed. This update also models the observation that false positives are rare, while false negatives are common when actions are executed on robots. Third, a robot moving between cells may receive sensor inputs relevant to the current task, e.g., it may unexpectedly have the target in its field of view. The robot, therefore, periodically processes input images at low resolution to update current beliefs. Fourth, robots learn and revise domain map using (laser) range data and acquire semantic labels (e.g., "kitchen" and "office") from humans.

Target objects in the experimental trials are assumed to be unique, and observations of targets are assumed to be independent of each other. In addition, although the incremental accumulation of evidence (of target occurrence) by the POMDP hierarchy can (in theory) be used to localize objects that move (or are moved), targets are assumed to stay within a local area, while they are being located by robots. These simplifications are used to focus on the underlying planning challenge and study the effects of factors such as prior knowledge and communication failures.

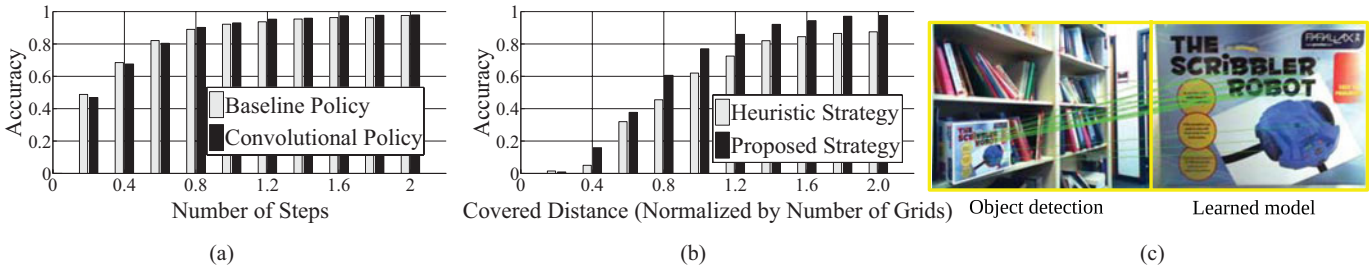

Fig. 3. (a) Target localization accuracy of the CC policy is similar to that of baseline policy. (b) CC policy enables the robot to find targets in a much smaller number of steps than a (greedy) heuristic policy. (c) Illustrative example of the use of BRIEF descriptor to represent and recognize objects.

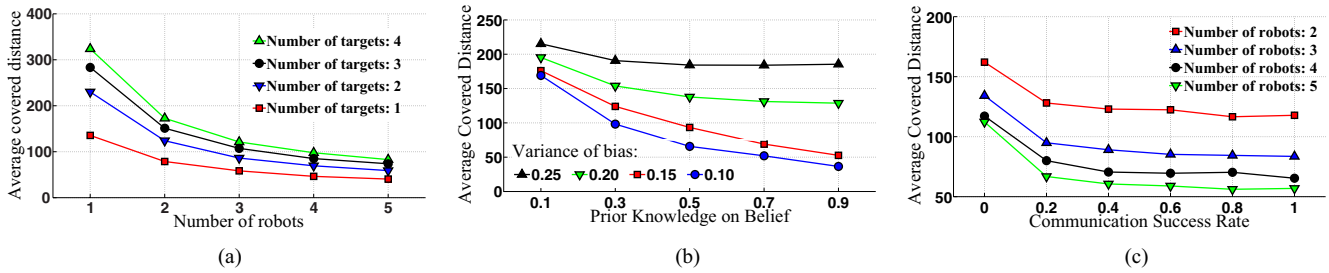

Fig. 4. (a) Belief merging and hierarchical POMDPs result in robust multirobot collaboration. (b) Performance improves if prior information is incorporated. (c) Performance is robust to dropped communication packages.

Experiments were designed to evaluate the following hypotheses: I) Constrained convolutional (CC) policy provides similar detection accuracy to nonconvolutional (i.e., baseline) policy but is much more efficient; II) CC policy significantly reduces time for reliable target localization compared with manually tuned heuristic search strategies; and III) belief merging enables a team of robots to fully utilize prior knowledge and collaborate, despite unreliable communication. Hypothesis I was evaluated in simulation, while hypotheses II and III were evaluated in simulation and on robots.

A. Simulation Experiments

In each simulated trial, a grid map of specific size was generated with the locations of targets and robots chosen randomly. A cell is assumed to contain a target when belief in the cell exceeds 0.9. Each data point in the following results is the average of 1000 simulated trials.

Hypothesis I was evaluated with the adaptive observation functions and policy reweighting described in (3)–(9). A baseline policy computed for a 5×5 map was used to extract a policy kernel that was used to compute policies for larger maps. Fig. 3(a) compares the CC policy against the baseline policy for a 7×7 map—the x-axis shows the number of times the policy was invoked, as a fraction of the number of states. A larger map was not used to generate the baseline policy (for comparison) because the time taken to generate a stable baseline policy increases exponentially. A trial was deemed successful if the target's location was identified correctly. No statistically significant difference was observed in the target localization accuracy obtained with CC and baseline policies.

Hypothesis II was evaluated by comparing the CC policy with a heuristic policy that makes greedy action choices. The results shown in Fig. 3(b) used a 15×15 convolutional policy generated from a 5×5 kernel. Existence of prior knowledge was simulated by adding bias to the initial belief—70% of the belief was uniformly distributed over all cells, while the remaining 30% was Gaussian distributed around the target. To generate the data points in Fig. 3(b), trials were terminated after a certain distance had been traveled—the cell with the largest belief was then considered the target's location. The robot's performance is scored as the weighted distance between the actual and detected locations of targets. Fig. 3(b) shows that the CC policy significantly reduces the distance traveled (and thus time taken) by the robot to locate targets with high accuracy.

Next, hypothesis III (i.e., multirobot collaboration capability) was evaluated. Assuming that all robots in a team move at the same speed, the average distance moved by robots in a team (in an episode or trial) was used as a measure of the team's performance; better collaboration will result in lower values of this measure. In each trial, robots and targets were placed randomly in a grid map. A Gaussian bias (20%) was added to the initial belief in a 3×3 area around every target, and the belief vector was normalized. To simulate unreliable communication, a *communication success rate* (CSR) parameter was introduced and set to 0.5, i.e., every other broadcasted package (on average) was not received.

Additional experiments were conducted to evaluate the effect of team size, prior knowledge, and CSR on multirobot collaboration. Fig. 4(a) shows the results for different numbers of robots and targets in a 15×15 grid map based on a real-world office scenario. The results show that the robots collaborate effectively to find the targets. Next, Fig. 4(b) shows the performance of a team of two robots tasked with localizing two targets, as a function of the bias in the initial belief. Robots are typically likely to have some prior knowledge of the locations of objects,

TABLE I
TARGET LOCALIZATION ACCURACY $\in [0, 1]$ AS A FUNCTION OF THE (NORMALIZED) DISTANCE TRAVELED BY THE TEAM OF ROBOTS

Approach	Normalized covered distance			
	0.5	1.0	1.5	2.0
Random	0.033	0.171	0.382	0.537
Heuristic	0.079	0.334	0.549	0.817
Proposed	0.153	0.544	0.825	0.957

The proposed approach enables accurate target localization in much less time than random and heuristic collaboration strategies.

(a) (b)

Fig. 5. Robot platforms used in experiments. (a) Erratic robot. (b) Nao robots.

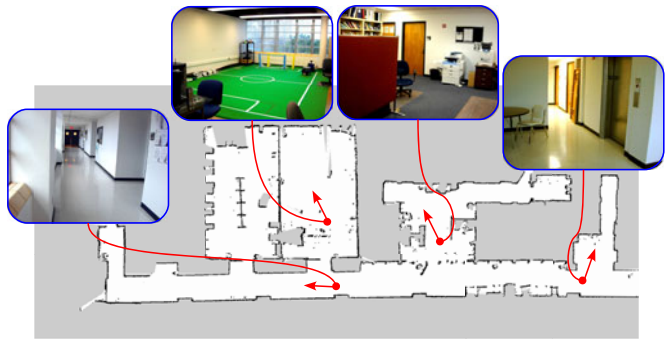

Fig. 6. Occupancy-grid map of the third floor of the CS department with 13 faculty offices, three research labs, a conference room, and a common area with a kitchen.

e.g., a microwave is likely to be found in the kitchen. Fig. 4(b) shows that robots are able to identify the targets faster as more information about target locations is made available or the information available about the targets' positions is more accurate (i.e., smaller variance of bias). Next, Fig. 4(c) summarizes results of experimental trials where the CSR was varied as robot teams were asked to localize two target objects. Communication between robots in the real world can be unreliable. The results in Fig. 4(c) indicate that although a low likelihood of successful communication (i.e., low CSR) hurts the team's performance, the time taken to localize targets stabilizes as CSR increases and is then no longer sensitive to the value of CSR.

Table I summarizes target localization accuracy $\in [0, 1]$ as a function of the distance traveled (normalized by the number of cells in the map), when two robots searched for targets in a 15×15 map. Initial positions of robots and targets were randomly assigned in each trial. The proposed approach (belief sharing with hierarchical POMDPs) was compared with 1) random selection of actions and assignment of targets to robots (row labeled "random") and 2) a (greedy) heuristic policy that selects targets and actions based on the cell with the largest belief (row labeled "heuristic"). To simulate realistic scenarios, prior belief was assigned to multiple areas in the map (including the target location). Results show that belief sharing in hierarchical POMDPs significantly reduces the distance traveled by robots to detect targets with high accuracy.

The experiments described above were repeated for different numbers of robots and targets, different levels of prior beliefs, and different values of CSR in domain maps ranging in size from 5×5 to 25×25. The experimental results indicate that using hierarchical POMDPs and the belief sharing strategy enables a team of robots to collaborate and localize target objects reliably and efficiently. As stated earlier, multiple robots are (intentionally) allowed to search for the same target, but instances of such overlap occur very infrequently.

B. Robot Experiments

Experiments were conducted on a wheeled robot and a team of humanoid robots in indoor domains (see Fig. 5).

1) Experiments on Wheeled Robot: The hierarchical POMDPs were used for planning sensing and information processing on the *Erratic* robot platform shown in Fig. 5(a). This robot is equipped with stereo and monocular cameras that provide 640×480 images at 30 Hz and a laser range finder with an angular range of $\pm 135^\circ$ for a distance of 30 m. All processing is performed using a dual-core 2.6-GHz processor on board the robot. Experiments were conducted in an indoor domain comprising multiple floors of the Computer Science Department at Texas Tech University. Fig. 6 shows the result of using a simultaneous localization and mapping algorithm [9] to learn the occupancy-grid map of a floor with three research labs, 13 faculty offices, a conference room, and a common area with a kitchen. The size of each cell is ≈ 2 m $\times$ 2 m, and the size of the learned policy kernel is 5×5.

Object models learned by the robot consist of color distributions and the Binary Robust Independent Elementary Features (BRIEF) [6], i.e., local image gradient features. Although BRIEF features are not rotation and scale invariant, images captured in the learning phase are automatically transformed to model different rotations and scales; features extracted from all these images populate object models. Such object models enable robots to recognize objects, despite partial occlusions. Fig. 3(c) shows a test image's BRIEF features being matched with those in a learned object model. Object models also include a measure of size to compute the relative distance and bearing of objects. Many scenes in this domain are cluttered, resulting in images with many ROIs and object models consisting of complex features. As stated in Section III-C, SP-POMDP has one (two) layer(s) for cluttered (uncluttered) scenes. During policy execution, the robot analyzes scenes from multiple viewpoints. Target objects include boxes, cups, books, and other robots in complex backgrounds. All targets are assumed to be visually distinguishable.

For modular software development, algorithms were implemented in the Robot Operating System (ROS) [26] framework.

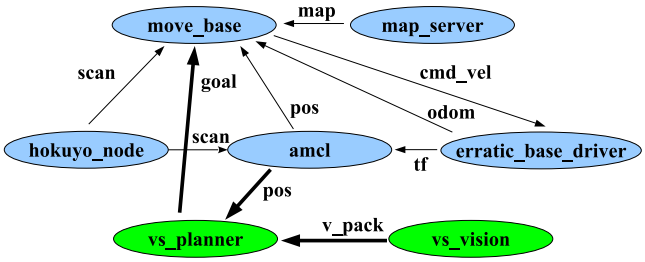

Fig. 7. Relevant subset of nodes in the ROS implementation of the proposed algorithms.

TABLE II
TARGET LOCALIZATION TIME EXPRESSED AS A FRACTION OF THE TIME TAKEN BY THE PROPOSED APPROACH

Search and Collaboration Strategies	Localization time for specific targets			
	Wheeled robot		Humanoid robots	
	Box in Fig. 3(c)	Nao robot	Boxes	Balls
Random	–	–	1.93	1.64
Heuristic	1.47	1.44	1.2	1.03
Proposed	1	1	1	1

Use of POMDP hierarchy enables the wheeled robot to identify targets reliably and efficiently. Sharing POMDP beliefs enables a team of humanoid robots to improve target localization time.

Fig. 7 gives an overview of a relevant subset of the ROS implementation. Our planning algorithms are placed in the *vs_planner* node, which communicates with the *vs_vision* node that processes input images to populate the <v_pack> package. This package contains the ID of detected objects, relative distance, and bearing of the objects, as well as (probability) measures of the certainty associated with the observations. Belief updates occur when the robot arrives at a desired cell and processes images of the scene, or processes images during navigation to a cell. The planner node sends coordinates of desired cells to the movement control node *move_base* and then waits for a response, e.g., `arrived` when the desired cell is reached or `not-arrived` when an unexpected change such as closing a door makes a location inaccessible. The *hokuyo_node* provides laser readings to the motion control node and the localization node *amcl*. The platform driver node *erratic_base_driver* moves the robot platform based on the velocity command `cmd_vel`. The `position` and `goal` are sent and received by nodes that aid in local path planning, localization, and navigation.

In each trial, the robot's initial position and the positions of target objects were chosen randomly. Belief distributions were initialized to give the robot some prior knowledge of object locations. The left half of Table II summarizes localization time for specific target objects (i.e., a representative subset of the experiments). Each data point is averaged over 10–15 trials. Since the robot and target positions differ between trials, results for *random* and *heuristic* strategies are expressed as a multiple of the *proposed* strategy's results, e.g., the average time taken to localize the *Box* with the POMDP hierarchy is 4.08 min. The heuristic strategy that makes greedy action choices requires significant manual tuning of the associated parameters. Table II does not show results for the *random* strategy due to the large variance; although the (average) multiplying factor is ≈ 3, many trials do not terminate even after 15 min. For all targets, the POMDP hierarchy significantly reduces the localization time, in comparison with random and heuristic (multiplying factor of 1.5) strategies. The results are more pronounced than in the simulation because the domain (see Fig. 6) is more complex. A video of an experimental trial can be viewed online at http://youtu.be/CbsC0ScuuBk

2) Experiments on Humanoid Robots: Multirobot collaboration experiments were conducted on the humanoid (Nao) robots shown in Fig. 5(b). The Nao is equipped with multiple monocular cameras that provide 640×480 images at 30 Hz and ultrasound sensors for obstacle avoidance. Since stable humanoid navigation on different surfaces is a challenge, experiments were conducted on an indoor (4 m $\times$ 6 m) robot soccer field, which is typically used by a team of Naos to play a competitive game of soccer. This moderately constrained domain captures the collaboration challenges we seek to address. Each robot has a domain map and localizes based on domain landmarks such as goals and field corners (with known map locations) detected in images. All scenes in this domain are treated as uncluttered, but challenging scenarios are created by artificially introducing obstacles that the robot(s) have to walk around to see targets and landmarks. All computation was performed using a 500-MHz processor on-board the robots. Robots broadcast packages to teammates to share information. The size of each cell in the domain map is $\approx$0.5 m, and the size of the learned policy kernel is 5×5.

Target objects include boxes and balls of different colors and shapes. Since objects are composed of homogeneous colors, gradient features are not used in the object models, and visual processing operators consist of algorithms that detect the dominant color and shape in each ROI. Scene processing was modeled as a two-layered POMDP, with a POMDP that selects operators (i.e., algorithms) to apply on each salient ROI in an image and a POMDP that controls the selection of image ROIs for processing. The transfer of control between SP-POMDP and VS-POMDP is described in Section III-C.

In all multirobot collaboration experiments, a team of (1–4) Naos localized target objects. The right half of Table II summarizes a representative subset of these experiments, where two Naos localized two targets (boxes and balls). The proposed strategy is compared with 1) a strategy that randomly assigns robots to targets and 2) a (greedy) heuristic collaboration strategy that requires substantial manual tuning to consider factors such as distance to target and presence of obstacles [30]. The target localization times are smaller due to collaborative effort and relative simplicity of the domain (compared with Fig. 6). Using the proposed strategy, the average time taken by two Naos to localize two boxes is 1.01 min. The proposed strategy significantly reduces the target localization time in comparison with the random strategy and performs better than (or at least as well as) the heuristic strategy. Similar results are obtained for different combinations (and numbers) of robots and targets. The effect of communication uncertainty is arbitrary with the heuristic strategy, e.g., robots cluster around targets and take twice as much time to localize targets. However, with the proposed collaboration strategy, delayed or lost communication packets did not affect target localization when CSR was above a low threshold (≈ 0.2). The results reported in Table II correspond to a

CSR of ≈ 0.5. Furthermore, adding or removing a team member resulted in the smooth redistribution of targets among robots. These experiments show that the POMDP hierarchy and belief sharing strategy enable robots to adapt visual sensing and information processing to the task at hand and collaborate despite unreliable communication.

V. Conclusion

This paper has described an algorithm for planning visual sensing, information processing, and collaboration in a team of robots. A hierarchical decomposition of POMDPs enables each mobile robot to automatically tailor visual sensing and information processing to each of a range of tasks at hand. The hierarchy incorporates CC policies and automatic belief propagation, enabling a robot to operate reliably and efficiently in complex indoor domains. Belief sharing between a team of robots is accomplished by augmenting the hierarchical POMDPs by a communication layer, enabling each robot to merge beliefs acquired by processing sensor inputs with the beliefs communicated by teammates. The robots are thus able to fully utilize the available information and collaborate in simulated and real-world domains.

One direction of further investigation is to explicitly model the sensing and actuation capabilities of different robots and incorporate these learned models to improve the multirobot collaboration capabilities. Experiments will also include a larger number of robots and targets, as well as different types of robots and information-processing algorithms. Furthermore, we are exploring the integration of the POMDP hierarchy with a knowledge representation and nonmonotonic logical inference paradigm [33]. The ultimate goal is to enable reliable, efficient, and autonomous multirobot (and human–robot) collaboration in complex real-world domains.

References

[1] A. Aydemir, M. Göbelbecker, A. Pronobis, K. Sjöö, and P. Jensfelt, "Plan-based object search and exploration using semantic spatial knowledge in the real world," presented at the Eur. Conf. Mobile Robots, Örebro, Sweden, Sep. 2011.

[2] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Math. Oper. Res.*, vol. 27, no. 4, pp. 819–840, Nov. 2002.

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

[4] O. Buffet and D. Aberdeen, "The factored policy-gradient planner," *Artif. Intell.*, vol. 173, nos. 5/6, pp. 722–747, 2009.

[5] N. J. Butko and J. R. Movellan, "I-POMDP: An infomax model of eye movement," in *Proc. IEEE Int. Conf. Develop. Learn.*, 2008, pp. 139–144.

[6] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2010, pp. 778–792.

[7] J. Capitán, L. Merino, F. Caballero, and A. Ollero, "Decentralized delayed-state information filter (DDSIF): A new approach for cooperative decentralized tracking," *Robot. Auton. Syst.*, vol. 59, no. 6, pp. 376–388, Jun. 2011.

[8] X. Chen, J. Ji, J. Jiang, G. Jin, F. Wang, and J. Xie, "Developing high-level cognitive functions for service robots," presented at the Int. Conf. Auton. Agents Multiagent Syst., Toronto, ON, Canada, May 10–14, 2010.

[9] G. Dissanayake, P. Newman, and S. Clark, "A solution to the simultaneous localization and map building (SLAM) problem," *Trans. Robot. Autom.*, vol. 17, no. 3, pp. 229–241, 2001.

[10] A. F. Foka and P. E. Trahanias, "Real-time hierarchical POMDPs for autonomous robot navigation," in *Proc. IJCAI Workshop Reason. Uncertainty Robot.*, 2005, pp. 561–571.

[11] C. Galindo, J.-A. Fernandez-Madrigal, J. Gonzalez, and A. Saffioti, "Robot task planning using semantic maps," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 955–966, 2008.

[12] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning: Theory and Practice*. San Francisco, CA, USA: Morgan Kaufmann, 2004.

[13] M. Göbelbecker, C. Gretton, and R. Dearden, "A switching planner for combined task and observation planning," presented at the 25th Conf. Artif. Intell., San Francisco, CA, USA, 2011.

[14] P. J. Gymtrasiewicz and P. Doshi, "A framework for sequential planning in multi-agent settings," *J. Artif. Intell. Res.*, vol. 24, pp. 49–79, 2005.

[15] L. Kaelbling, M. Littman, and A. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, pp. 99–134, 1998.

[16] L. Kaelbling and T. Lozano-Perez, "Domain and plan representation for task and motion planning in uncertain domains," in *Proc. IROS Worksh. Knowl. Represent. Auton. Robots*, 2011.

[17] A. Krause, A. Singh, and C. Guestrin, "Near-optimal Sensor Placements in Gaussian Processes: Theory, efficient algorithms and empirical studies," *J. Mach. Learn. Res.*, vol. 9, pp. 235–284, 2008.

[18] J. Y. Kwak, R. Yang, Z. Yin, M. Taylor, and M. Tambe, "Teamwork and coordination under model uncertainty in DEC-POMDPs," in *Proc. AAAI Worksh. Interact. Decis. Theory Game Theory*, 2010, pp. 30–36.

[19] L. Li, V. Bulitko, R. Greiner, and I. Levner, "Improving an adaptive image interpretation system by leveraging," in *Proc. Australian New Zealand Conf. Intell. Inf. Syst.*, 2003.

[20] A. Makarenko, A. Brooks, S. Williams, and H. Durrant-Whyte, "A decentralized architecture for active sensor networks," in *Proc. Int. Conf. Robot. Autom.*, 2004, pp. 1097–1102.

[21] S. C. Ong, S. W. Png, D. Hsu, and W. S. Lee, "Planning under uncertainty for robotic tasks with mixed observability," *Int. J. Robot. Res.*, vol. 29, no. 8, pp. 1053–1068, Jul. 2010.

[22] L. Panait and S. Luke, "Cooperative multi-agent learning: the state of the art," *J. Auton. Agents Multiagent Syst.*, vol. 11, no. 3, pp. 387–434, Nov. 2005.

[23] L. E. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems," *J. Phys. Agents (Invited Article) Special Issue Multi-Robot Syst.*, vol. 2, no. 2, pp. 5–14, 2008.

[24] R. Petrick and F. Bacchus, "Extending the knowledge-based approach to planning with incomplete information and sensing," in *Proc. Int. Conf. Automat. Plann. Schedul.*, 2004, pp. 2–11.

[25] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun, "Towards robotic assistants in nursing homes: challenges and results," *Robot. Autonom. Syst. (Special Issue on Socially Interactive Robots)*, vol. 42, pp. 271–281, 2003.

[26] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *Proc. ICRA Worksh. Open Source Softw.*, 2009.

[27] P. E. Rybski, A. Larson, H. Veeraraghavan, M. LaPoint, and M. Gini, "Communication strategies in Multi-robot search and retrieval: Experiences withMinDART," in *Proc. Int. Symp. Distrib. Auton. Robot. Syst.*, 2004, pp. 301–310.

[28] S. Seuken and S. Zilberstein, "Improved memory-bounded dynamic programming for decentralized POMDPs," presented at the Int. Conf. Uncertainty Artif. Intell., Vancouver, BC, Canada, 2007.

[29] M. Sridharan, J. Wyatt and R. Dearden, "Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs," *Artif. Intell.*, vol. 174, pp. 704–725, 2010.

[30] P. Stone, M. Sridharan, D. Stronger, G. Kuhlmann, N. Kohl, P. Fidelman, and N. K. Jong, "From pixels to multi-robot decision-making: A study in uncertainty," *Robot. Auton. Syst. (Special issue Plann. Under Uncertainty Robot.)*, vol. 54, no. 11, pp. 933–943, 2006.

[31] G. Theocharous, "Hierarchical learning and planning in partially observable Markov decision processes," Ph.D. dissertation, Michigan State Univ., East Lansing, MI, USA, 2002.

[32] S. Zhang and M. Sridharan, "Active visual sensing and collaboration on mobile robots using hierarchical POMDPs," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2012, pp. 181–188.

[33] S. Zhang and M. Sridharan, F. S. Bao, "ASP+POMDP: Integrating non-monotonic logic programming and probabilistic planning on robots," in *Proc. Int. Conf. Development Learn.*, 2012, pp. 1–7.

Shiqi Zhang received the B.S. and M.S. degrees from the Harbin Institute of Technology, Harbin, China, in 2006 and 2008, respectively. He is currently working toward the Doctoral degree with the Department of Computer Science, Texas Tech University, Lubbock, TX, USA.

He is a member of the Stochastic Estimation and Autonomous Robotics Laboratory, Texas Tech University. He was a Research Intern at Microsoft Research Asia in Summer 2012. His research interests include probabilistic planning and logical reasoning on mobile robots, mobile sensing, and machine learning.

Mohan Sridharan received the Ph.D. degree in electrical and computer engineering from the University of Texas at Austin, Austin, TX, USA, in 2007.

He is currently an Assistant Professor of computer science with Texas Tech University, Lubbock, TX, USA, where he directs the Stochastic Estimation and Autonomous Robotics Laboratory. He was a Research Fellow with the School of Computer Science, University of Birmingham, Birmingham, U.K., working on the European Union Cognitive Systems Project, between August 2007 and October 2008. His current research interests include machine learning, planning, computer vision, and cognitive science as applied to autonomous mobile robots and intelligent agents.

Christian Washington is currently an Undergraduate Student in computer engineering with Louisiana State University, Baton Rouge, LA, USA.

He is also a member of the Tangible Visualization Research Group, Center for Computation and Technology, Louisiana State University. In Summer 2012, he was a Research Intern supported by the National Science Foundation Research Experiences for Undergraduates program at Texas Tech University, Lubbock, TX, USA. His research interest is in the area of autonomous systems.