In *IEEE International Conference on Robotics and Automation* (ICRA 11),
Shanghai, China, May 9-13 2011.

To Look or Not to Look: A Hierarchical Representation for Visual Planning on Mobile Robots

Shiqi Zhang
Department of Computer Science
Texas Tech University, TX 79409
s.zhang@ttu.edu

Mohan Sridharan
Department of Computer Science
Texas Tech University, TX 79409
mohan.sridharan@ttu.edu

Xiang Li
Department of Computer Science
Texas Tech University, TX 79409
xiang.li@ttu.edu

***Abstract*— Mobile robots are increasingly being used in real-world applications due to the ready availability of high-fidelity sensors and the development of sophisticated information processing algorithms. However, one key challenge to the widespread deployment of mobile robots equipped with multiple sensors and processing algorithms is the ability to autonomously tailor sensing and information processing to the task at hand. This paper poses this challenge as the task of planning under uncertainty, and more specifically as an instance of probabilistic sequential decision-making. A novel hierarchy of partially observable Markov decision processes (POMDPs) is incorporated, which uses constrained-convolutional policies and automatic belief propagation to achieve efficient and reliable operation on mobile robots. All algorithms are implemented and evaluated on simulated and physical robot platforms for the task of searching for target objects in dynamic indoor environments.**

Keywords: Planning, Scheduling and Coordination; Computer Vision for Robotics and Automation; Wheeled Robots.

I. INTRODUCTION

Mobile robots are increasingly playing an important role in practical applications due to developments in sensor technology and sensory input processing algorithms [5], [25]. However, real-world application domains are characterized by partial observability (i.e., the state of the system is not directly observable) and non-determinism (i.e., actuation and sensing are unreliable), and algorithms with different reliabilities and computational complexities can be used to process the sensory inputs. The challenges are all the more pronounced with visual input because of the inherent sensitivity to environmental changes and the high computational complexity of the corresponding processing algorithms. As a result, despite being a rich source of information, vision is still not fully exploited in robot applications. Widespread deployment of mobile robots in the real-world requires autonomous operation and full utilization of the relevant sensory inputs. One solution is to retain capabilities to support a wide variety of tasks, and automatically choose the relevant subset based on the task at hand. This paper presents such an approach for visual sensing and information processing, i.e., for visual processing management.

Planning under uncertainty can be posed as an instance of probabilistic sequential decision-making, and more specifically, using partially observable Markov decision processes (POMDPs) [12]. However, POMDP formulations of practical domains soon become intractable due to the associated state space explosion and computational complexity, even when approximate solvers are used [21]. This paper proposes a hierarchical POMDP, whose layers jointly decide: *where to look?*, *what to process?* and *how to process?* based on the task at hand. Our prior work used an instance of the intermediate and lower levels of the hierarchy to enable a robot to plan visual processing in a tabletop scenario [22]. This paper extends this approach to enable a mobile robot to direct sensing to locations that are relevant to the task, in addition to considering the reliability and complexity of the processing operators to determine the sequence of operators most suitable for a given task. Constrained convolutional policies are used to exploit the local invariance of visual sensing, while learned sensor models and belief propagation between the levels of the hierarchy are used to generate POMDP models automatically, thereby resulting in reliable and efficient operation. All algorithms are evaluated on simulated and physical robots for the task of locating target objects in dynamic indoor domains.

The remainder of the paper is organized as follows. Section II briefly describes related work, while the proposed hierarchical planning scheme is described in Section III. The experimental setup and results are presented in Section IV, followed by conclusions in Section V.

II. RELATED WORK

Classical planning research has produced many methods for planning a pipeline of operators for a high-level goal. Many such methods use deterministic models, where the preconditions and effects of actions are propositions that are required to be true a priori, or are made true by operator execution [9]. However, robot domains are characterized by partial observability and non-determinism. Recent research has hence focused on relaxing some of the limitations of classical planners. The PKS planner [19] 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 state of the world is determined uniquely by the actions performed, but the agent's knowledge of that state is not. Continual Planning (CP) [2] interleaves planning, execution and monitoring, and postpones reasoning about uncertain states, asserting that action preconditions will be met when that point is reached during plan execution. If the preconditions are not met during execution, or are met earlier, replanning occurs. In robot domains, it may be necessary to accumulate evidence by applying operators more than once, which cannot be modeled using PKS or CP.

In vision research, image interpretation has been modeled as an MDP or POMDP. For instance, Li et al. [15] use human-annotated images to determine the reward structure, explore the state space and compute value functions in an MDP used for image interpretation. Online operation involves action choices that maximize the learned functions. Sensor placement and information processing have also been posed as an active sensing problem [14]. Sensor placements in spatial phenomena have also been modeled as Gaussian processes using submodular functions [13]. However, many visual planning tasks are not submodular, and modeling probability densities using manual feedback and extensive trials on mobile robots is infeasible.

Though a POMDP formulation is well-suited for domains with partially observability and unreliable actions, the state space explosion makes it intractable for most practical domains. Hierarchical approaches have hence been proposed for behavior control and navigation tasks on robots, with the top level action being a collection of simpler actions modeled as smaller POMDPs [8], [20] but a significant amount of data for the hierarchy and model creation is typically hand-coded. Recent work has also focused on learning POMDP observation models [1]; using information maximization for POMDP-based visual search [4]; developing faster solution methods or discovering the hierarchy [21], [24], resulting in real-world applications [10], [11]. However, these methods are computationally expensive or unable to create the POMDP models automatically. This paper presents a novel POMDP hierarchy that enables a robot to automatically tailor visual sensing and processing in dynamic domains.

III. PROBLEM FORMULATION

This section first describes the problem domain, followed by the proposed hierarchical POMDP formulation.

A. Domain Description

Figure 1(a) describes the long-term goal of this project: creating autonomous robots that can assist humans in dynamic applications such as health care. Achieving this goal requires, among other things, autonomous learning and processing management. This paper addresses autonomous visual processing management using hierarchical POMDPs. Consider, for instance, the task of finding an object (e.g., a humanoid). The intuitive approach would be to first decide on the 3D scene likely to contain the target object. The robot can then capture images of the scene, such as the one shown in the top right of Figure 1(a). The images can be processed to obtain salient *regions of interest* (ROIs). The robot can process such ROIs using any of the available visual operators for tasks such as segmentation, object recognition and scene reconstruction. This paper models sensing and information processing as actions, and uses the terms "operators" and "actions" interchangeably. The goal is to analyze a subset of ROIs in a sequence of images of a sequence of scenes most relevant to the task at hand. The experimental platforms are the humanoid robot [18] and the wheeled robot [6] in Figure 1(c), which are equipped with cameras, range finders, on-board processing (500MHz, 2.2GHz) and wi-fi capabilities, in addition to a range of algorithms for extracting information from different sensory inputs.

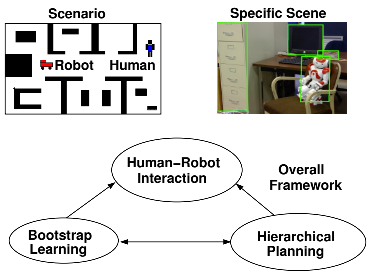
B. Hierarchical POMDPs

As shown in Figure 1(b), the three levels of the proposed POMDP hierarchy match the *functional* requirements of visual processing. The high-level POMDP ("HL-POMDP") chooses a sequence of 3D scenes to process based on the task (*where to look?*). The intermediate-level POMDP ("IL-POMDP") analyzes images of a chosen scene to select the salient region of interest (ROI) in the image to be processed next (*what to process?*). Finally, each ROI is modeled using a lower-level POMDP ("LL-POMDP") that computes the sequence of visual operators to be applied in order to address the specific task (*how to process?*). Our prior work used an instance of the intermediate and lower levels of the hierarchy to enable human-robot interaction in a tabletop scenario [22]. However, target objects in real-world domains, e.g., an office, can exist in different locations within a room or in different rooms. The robot has to move to analyze different scenes and locate the object. The work described in this paper builds on prior work to enable a mobile robot to locate desired objects in complex indoor environments. The description below focuses on the HL visual sensing, while more details on the IL-POMDPs and LL-POMDPs are in [22].

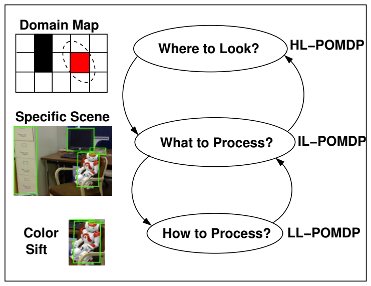
Consider the task of locating a target object. Assume that the robot has learned an environmental map using a Simultaneous Localization And Mapping (SLAM) algorithm [7]. The 3D area is then represented as a 2D *occupancy grid*—top left of Figure 1(b). Each grid-cell is associated with a probability that represents the likelihood of occurrence of the target object. The HL-POMDP poses sensing as the task of maximizing information gain, i.e., reducing the entropy in the belief state. For a grid with N cells, the HL-POMDP tuple $\langle \mathcal{S}^H, \mathcal{A}^H, \mathcal{T}^H, \mathcal{Z}^H, \mathcal{O}^H, \mathcal{R}^H \rangle$ is defined as:

- $\mathcal{S}^H : s_i, i \in [1, N]$ is the state vector; s_i corresponds to the event that a target is in grid-cell i. The assumption is that an object only exists in one grid-cell at a time.
- $\mathcal{A}^H : a_i, i \in [1, N]$ is the set of actions, where a_i corresponds to the event that the robot moves to to grid-cell i and analyzes the corresponding scene.
- $\mathcal{T}^H : \mathcal{S}^H \times \mathcal{A}^H \times \mathcal{S}'^H \rightarrow [0, 1]$ is the state transition function. It is an identity matrix for actions that only observe the state.
- $\mathcal{Z}^H : \{\text{present, absent}\}$ is the set of observations, indicating the presence or absence of the target in the cell being analyzed.
- $\mathcal{O}^H : \mathcal{S}^H \times \mathcal{A}^H \times \mathcal{Z}^H \rightarrow [0, 1]$ is the observation function. It is computed automatically, as described below.
- $\mathcal{R}^H : \mathcal{S}^H \times \mathcal{A}^H \rightarrow \Re$ is the reward specification, which is based on belief entropy as described below.

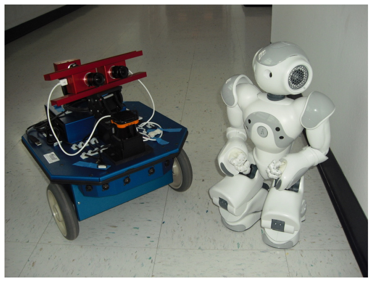
Since the true state of the system cannot be observed, the robot maintains a probability distribution over the state (*belief state*). For belief state B_t at time t, reward $\mathcal{R}^H$ of an



(a) Framework Overview.



(b) Proposed Hierarchy.



(c) Robot platforms.

Fig. 1. Visual processing management: (a) Overall scenario and a sample image with regions-of-interest (ROIs) extracted and bounded by rectangles; (b) The proposed planning hierarchy; (c) Robot platforms.

action is defined as the InfoMax objective function [4], i.e., the reduction in entropy in the resultant belief state B_{t+1}. The *entropy* of the belief distribution B_t can be defined as:

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

where b^i is the i^{th} entry of the belief state distributed over the map with N cells. The reward function is defined as:

$$\begin{aligned}\mathcal{R}^H(a_{t,i}) &:= \mathcal{H}(B_{t-1}) - \mathcal{H}(B_t) \qquad (2)\\ &= \sum_k b_t^k log(b_t^k) - \sum_j b_{t-1}^j log(b_{t-1}^j)\end{aligned}$$

When nothing is known about the target's location, the belief is uniformly distributed and entropy is maximum. As the belief distribution converges to states likely to be target locations, the entropy progressively reduces. Next, the observation function is defined based on the expected performance of the lower levels of the hierarchy:

$$\begin{aligned}\mathcal{O}(z_i, s_j, a_k) = Pr(z_i = present|s_j, a_k) = \eta \cdot e^{-\frac{\lambda\mu^2}{2\sigma^2}} \qquad (3)\\ \mu = f_\mu(s_j, a_k), \quad \sigma^2 = f_{\sigma^2}(\mathcal{O}, \mathcal{O}^I|s_j, a_k)\end{aligned}$$

where the probability of a particular observation in cell i given that the target is in cell j and the focus is on cell k, i.e., $p(z_i|s_j, a_k)$, is a Gaussian, whose mean depends on the target location, the grid-cell being examined and the camera's field of view. The variance of the Gaussian is based on the observation functions of the lower levels of the hierarchy $(\mathcal{O}, \mathcal{O}^I)$. This formulation of the observation function models the fact that a robot can detect a target with the highest likelihood when it is close to the target. The belief update then proceeds as follows:

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

POMDP solvers use such a model to compute a *policy*: $\pi^H : B_t \mapsto a_{t+1}$ that maps belief states to action choices. The computed policy has to minimize the entropy in B_t over a planning horizon of T steps. In this paper, policy gradient methods in the LibPG library [3] are used to compute the HL policy. The policy is hence a probability distribution over possible actions in the form of "weights" that are used to choose the *best* action for a given belief state.

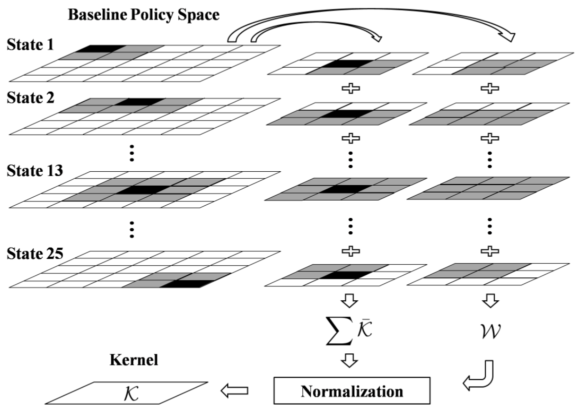


Fig. 2. Extract a 3×3 kernel from a 5×5 baseline policy.

1) Convolutional Kernels: In a practical domain, the number of grid-cells can be quite large and the environment can change dynamically. The exponential increase in state space and the worst-case exponential time complexity of POMDP solvers can pose a formidable challenge to solving the associated HL-POMDP. The proposed approach therefore learns a convolutional policy kernel that exploits the rotation and shift-invariance of visual search:

$$\bar{\mathcal{K}}(s) = (\pi^H \otimes \mathcal{C}_m^K)(s) = \int \pi^H(\tilde{s})\mathcal{C}_m^K(s - \tilde{s})d\tilde{s} \tag{5}$$

$$\mathcal{K} = (\sum_{a_i} \bar{\mathcal{K}}) \cdot / \mathcal{W}$$

where π^H is the HL-POMDP policy, $\bar{\mathcal{K}}$ is the un-normalized kernel, $\mathcal{K}$ is the normalized kernel and $\mathcal{C}_m^K$ is the convolution operator whose size decides the size of target kernel.

In Figure 2, a 3×3 policy kernel is extracted from a 5×5 baseline policy π^H that is in the form of a 2D matrix whose rows and columns correspond to specific states and actions. Each row is re-arranged to obtain a 2D matrix of the same size as the map—this matrix stores action weights when focusing on a specific state. The policy is hence decomposed into layers—see the leftmost column of Figure 2. When a robot visits a specific grid-cell, only the beliefs immediately around that grid-cell change. The proposed approach hence learns a *policy kernel* that focuses on a local area and sets all other weights to a much smaller value. A 3×3 policy kernel $\bar{\mathcal{K}}$ is computed by convolving a 3×3 mask $\mathcal{C}_m^K$ with these policy layers, taking into account the weights in the region

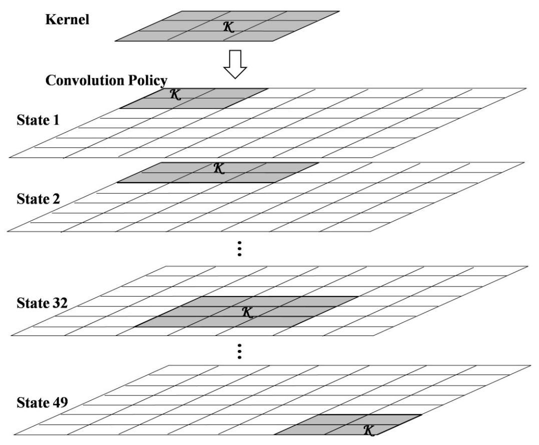


Fig. 3. Extend a 3×3 kernel into a 7×7 policy by convolution.

covered by the mask—middle column of Figure 2. Since the weights of the grid-cells outside the masked region are not taken into account, the resultant kernel is not normalized. The number of accumulated weights for each action is hence counted and the matrix $\mathcal{W}$, as shown in the right column of Figure 2, is used to obtain the normalized kernel $\mathcal{K}$.

The computed kernel does not assign action weights to the grid-cells away from the center of the convolution mask. Since these weights are usually much lower than the values in the kernel, they are all set to a small value:

$$\mathcal{W}^B = \frac{\sum\limits_{actions}\sum\limits_{states}\pi^H - \sum\limits_{actions}\sum\bar{\mathcal{K}}}{N_{actions}\times N_{states} - \sum\mathcal{W}} \tag{6}$$

where $N_{actions}$ and N_{states} are the number of actions and states respectively. When the learned policy kernel is used to generate policies for larger maps, the number of states covered by the kernel remains unchanged. The value of $\mathcal{W}^B$ is hence revised using a heuristic function, such that the ratio of belief distributions over the area covered and uncovered by the kernel is similar over different maps:

$$\hat{\mathcal{W}}^B = \mathcal{W}^B - ln(\frac{N^E_{states} - \sum\mathcal{W}}{N^B_{states} - \sum\mathcal{W}}) \tag{7}$$

where, N^E_{states} and N^B_{states} are the number of states in the large map and baseline kernel respectively,

2) Policy Extension: The policy kernel is used to efficiently compute the convolutional policy for a larger map:

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

where π^H_C is the convolutional policy, $\mathcal{K}$ is the policy kernel and $\mathcal{C}^E_m$ is the convolution mask of the same size as the target map. Consider Figure 3, where the policy for a 7×7 map is computed by convolving the 3×3 kernel with a 7×7 mask $\mathcal{C}^E_m$. The desired policy is generated one layer at a time, by centering the kernel on the corresponding state. In each of the 49 layers of the current example, values outside the 3×3 kernel are assigned the base weight of Equation 7.

A mobile robot has to physically move between grid-cells. Since this movement is unreliable, it is associated with a heuristic cost proportional to the distance to be moved. During policy execution, each action's weights are hence revised based on distance, grid-cell size and speed:

$$\hat{w}(i) = \frac{w(i)\frac{1}{1+\frac{dist(a_j,a_i)}{\text{grid-size}\times\text{speed}}}}{\text{normalizer}} \tag{9}$$

where $dist(a_j, a_i)$ is the distance between the current grid-cell and the candidate grid-cell. The weight of a cell decreases as the time (to travel to that cell) increases. The modified policy selects the grid-cell (i.e., 3D scene) suitable for analysis by trading off the likelihood of locating the target against the cost of traveling to that location. Once an action is selected, the robot moves to this location, and analyzes images of this scene using the IL-POMDP and LL-POMDPs.

Consider the task of locating a target object in an indoor domain. The learned map of the domain is used to generate the HL-POMDP, which is solved to obtain the HL policy for the desired target object. The HL policy is used to choose an appropriate scene of the environment to analyze. The robot moves to this scene, captures an image of the scene and detects salient regions of interest (ROIs) in the image. Each image ROI is modeled as an LL-POMDP, where the action choices are the specific information processing operators (e.g., detect color or shape). The corresponding LL policy provides the best sequence of operators to apply on the specific ROI to investigate the presence of the target object. The LL policies are used to automatically generate an IL-POMDP that controls the processing of all the image ROIs. The IL-POMDP is solved to get the IL policy, each of whose actions directs the robot's attention to a specific ROI. The presence or absence of the object in the ROI is analyzed using the corresponding LL policy and the result is used by the IL policy to update the belief across all image ROIs and choose a ROI to analyze further. The IL policy terminates when the presence (in a specific ROI) or absence of the target object in the image is computed. This information is used by the HL policy for a belief update over the 2D grid map, and the subsequent action choice leads to a similar analysis of another 3D scene. The process terminates when the target object is found with high confidence or a time threshold is exceeded. The key fact is that the hierarchical scheme results in reliable, efficient and autonomous operation.

IV. EXPERIMENTAL SETUP AND RESULTS

This section describes the experimental setup and discusses the experimental results.

A. Experimental Setup

In the experiments, the humanoid robot in Figure 1(c) was used as the target object, characterized by color distributions and local gradient features [17] extracted using a modified version of the *VLFeat* [26] open-source library. The object models were learned using an algorithm developed by our research group [16]. All algorithms were evaluated in simulation and on robots in complex office environments.

Given the focus on evaluating visual search, the individual steps in the IL and LL-POMDPs are not described below.

When the HL policy made an action choice, the result of applying the learned object models for target recognition in the corresponding image ROIs was directly used for belief updates. In addition, the obstacles and constraints present in the dynamic environment were used to suitably modify the HL policy computed from the convolutional kernel.

When the robot detects a target, it also computes the relative distance and bearing of the target. However, including the orientation in the observation set will increase the complexity of the observation model and negate the local invariance in policy space. The belief update was therefore modified based on target presence or absence:

$$\text{if absent} \qquad (10)$$

$$b(s') = \frac{O(s',a,o)\sum_{s\in S} T(s,a,s')b(s)}{Pr(o|a,b)} = \frac{O(s',a,o)b(s)}{Pr(o|a,b)}$$

else

$$b(s') = \frac{O(s',\hat{a},o)\sum_{s\in S} T(s,\hat{a},s')b(s)}{Pr(o|\hat{a},b)} = \frac{O(s',\hat{a},o)b(s)}{Pr(o|\hat{a},b)}$$

where, $b(s')$ represents the updated belief state after action a. Since the transition functions are identity matrices, the update equation can be simplified as shown. When a target is detected, the relative distance and bearing are used to find the global location of the target in the grid map (based on robot's localization) and the belief update proceeds as if the action corresponding to this global location had been executed: $\hat{a}$. This update scheme, known as *directed re-weighting*, was used in all experiments conducted on robot platforms.

B. Simulation Experiments

Simulation experiments were included because of the intractability of executing many trials on the physical robot. The simulator is realistic because it uses the same model parameters computed for the physical robot—Section III-B. In each simulated trial, a grid map of a specific size was generated with the locations of the target object and the robot chosen randomly. Then, the ability of the robot to detect the target with different initial beliefs was analyzed.

A baseline policy was computed for a relatively small grid map: 5×5, from which the 3×3 policy kernel was derived. The kernel computation is an one-time process. Though the POMDP model for this grid has only 25 states, it takes ≈ 5 hours to get an acceptable policy with the average reward still growing. Using the convolutional policy hence provides significant benefits for larger maps.

First, the constrained convolutional (CC) policy was evaluated against the baseline (i.e., non-convolutional) policy on simulated grids. Over a set of 1000 trials (with targets at random locations), the CC policy's detection accuracy was similar to the baseline policy. A trial was deemed successful if the target was identified in the correct grid-cell. Figure 4(a) shows the results for a 5×5 grid—the x-axis shows the number of times the policy was invoked, as a fraction of the number of states. Unlike the baseline policy, the CC policy was computed in no time from the 3×3 kernel.

The convolutional policy's performance was then compared against a policy that generates random actions, as a function of the number of actions the robot is allowed to execute (expressed as a fraction of the number of states). These experiments used a 15×15 convolutional policy generated from a 3×3 kernel. As before, the location of the robot and the target were randomly selected. In order to simulate prior knowledge of target location, 70% of the belief was uniformly distributed over all grids, and 30% of the belief was Gaussian-distributed surrounding the target. Each point in Figure 4(b) is the average of 1000 trials. At the end of each trial, the belief vector entry with the largest value was taken as the target location. The robot's performance was scored as the weighted distance between the actual target location and the detected location. The convolutional policy was observed to greatly reduce the number of steps taken to compute the target's location. The same CC policy was also used to compare directed re-weighting (Equation 10) against the policy execution without the special belief update scheme. Figure 4(c) shows that the directed re-weighting provides better performance despite added noise in the distance and bearing measurements.

TABLE I
AVERAGE NUMBER OF STEPS TO ACHIEVE ACCURACY OF 0.95

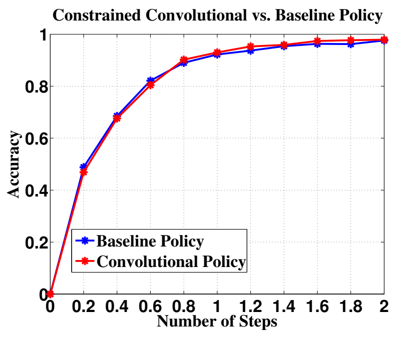
Bias		Covariance	
	0.1	0.2	0.3
10%	0.877	1.132	1.357
30%	0.521	1.019	1.274
50%	0.462	1.000	1.236

The next experiment computed the number of actions required to achieve a high detection accuracy (0.95), as a function of the initial bias and variance. Table I reports the average number of action steps as a fraction of the total number of states. As expected, a smaller number of actions are required to find a target with a larger initial bias. In addition, if the initial bias has a larger variance, a larger number of actions are required to find the target.

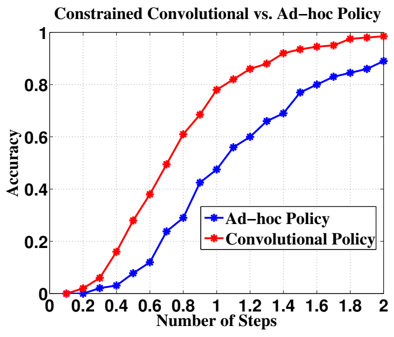
C. Implementation on robots

The algorithms were also evaluated on the robot platforms of Figure 1(c). When the robot arrives at a specific grid-cell, it processes a subset of ROIs of images of that scene by extracting visual features from the ROIs and comparing them with the learned object models. If no prior information is available, the robot initially starts off with a random search in nearby grid-cells. If some evidence of target presence in a specific grid-cell is obtained, the robot checks that grid-cell carefully to eliminate the effect of spurious observations. In the real-world, the robot almost always has some prior knowledge of possible target locations. This information is exploited by the proposed planning scheme by incorporating a bias in the initial belief, resulting in reliable and efficient recognition of the desired target.

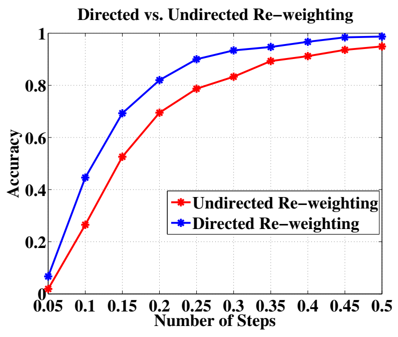
Next, the constrained convolutional policy was compared with an ad-hoc policy on different grid-sizes. The ad-hoc policy results in probabilistic updates, and a search based on manually generated heuristics and some random actions [23]. Figure 4(b) summarizes the accuracy of performance over 1000 trials in simulation and 30 trials on the robot in indoor



(a) CC vs. Baseline.



(b) CC vs. Ad-hoc.



(c) Directed Re-weighting.

Fig. 4. (a) Constrained convolutional policies provide performance similar to an expensive baseline policy; (b) Convolutional policies performs better than a ad-hoc search strategy; (c) Accuracy comparison from simulated trials and directed re-weighting.

corridors and offices that are mapped to a 15×15 grid—the individual grid-cells vary in size from 1m to 3m depending on the field of view of the associated cameras. As stated earlier, some simulation experiments were run with various levels of added noise. The CC policy is consistently much better than the ad-hoc policy. The errors in the CC policy correspond to cases where the target object is close to the edge between two grid-cells. On the physical robots, the availability of different 3D views ensures that the target is recognized in the correct location. Over extensive trials with different grid-sizes (3×3 to 25×25), the CC policy is more reliable and efficient than the ad-hoc strategy. Though the reduction in planning time in comparison to the baseline method depends on the size of the grid, the CC policy always leads to the desired real-time operation. Furthermore, the CC policy results in an (average) accuracy of 96% in comparison to the 80% accuracy of the ad-hoc strategy.

V. CONCLUSION

This paper described a novel hierarchical planning scheme that enables a mobile robot to tailor visual sensing and information processing to the task at hand. The key contributions are the constrained convolutional policies that exploit the local invariance of visual search, automatic belief propagation between the levels of the hierarchy, and the learned models of the performance of the visual operators. As a result, the robot is able to sequence a subset of the existing (unreliable) visual sensing and information processing operators to accomplish the task at hand. Future research will integrate operators that change the physical state of the system, and enable the simultaneous identification of multiple targets in complex domains by a team of autonomous mobile robots.

REFERENCES

[1] A. Atrash and J. Pineau. A Bayesian Method for Learning POMDP Observation Parameters for Robot Interaction Management Systems. In *The International POMDP Pratitioners Workshop*, 2010.
[2] M. Brenner and B. Nebel. Continual Planning and Acting in Dynamic Multiagent Environments. *JAAMAS*, 2008.
[3] O. Buffet and D. Aberdeen. The Factored Policy-Gradient Planner. *Artificial Intelligence*, 173(5-6):722–747, 2009.
[4] N. J. Butko and J. R. Movellan. I-POMDP: An Infomax Model of Eye Movement. In *ICDL*, 2008.
[5] J. Casper and R. R. Murphy. Human-robot interactions during urban search and rescue at the wtc. In *Transactions on SMC*, 2003.
[6] Videre Design. Videre Design Robot and Sensors, 2010. `http://www.videredesign.com/index.php?id=21`.
[7] G. Dissanayake, P. Newman, and S. Clark. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.
[8] A. F. Foka and P. E. Trahanias. Real-time Hierarchical POMDPs for Autonomous Robot Navigation. In *IJCAI Workshop on Reasoning with Uncertainty in Robotics*, 2005.
[9] M. Ghallab, D. Nau, and P. Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, 2004.
[10] J. Hoey, A. Monk, and A. Mihailidis. People, Sensors,Decisions: Customizable and Adaptive Technologies for Assistance in Healthcare. In *POMDP Pratitioners Workshop*, 2010.
[11] K. Hsiao, L. P. Kaelbling, and T. Lozano-Perez. Grasping POMDPs. In *International Conference on Robotics and Automation*, 2007.
[12] L. Kaelbling, M. Littman, and A. Cassandra. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101:99–134, 1998.
[13] A. Krause, A. Singh, and C. Guestrin. Near-optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research*, 9:235–284, 2008.
[14] C. Kreucher, K. Kastella, and A. Hero. Sensor Management using An Active Sensing Approach. *IEEE Transactions on Signal Processing*, 85(3):607–624, 2005.
[15] L. Li, V. Bulitko, R. Greiner, and I. Levner. Improving an Adaptive Image Interpretation System by Leveraging. In *Australian and New Zealand Conference on Intelligent Information Systems*, 2003.
[16] X. Li and M. Sridharan. Safe Navigation on a Mobile Robot using Local and Temporal Visual Cues. In *International Conference on Intelligent Autonomous Systems*, 2010.
[17] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
[18] Nao. The Aldebaran Nao Robots, 2008. `http://www.aldebaran-robotics.com/`.
[19] R. Petrick and F. Bacchus. Extending the Knowledge-Based approach to Planning with Incomplete Information and Sensing. In *ICAPS*, pages 2–11, 2004.
[20] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun. Towards Robotic Assistants in Nursing Homes: Challenges and Results. In *RAS Special Issue on Socially Interactive Robots*, 2003.
[21] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online Planning Algorithms for POMDPs. *JAIR*, 32:663–704, 2008.
[22] M. Sridharan, J. Wyatt, and R. Dearden. Planning to See: A Hierarchical Aprroach to Planning Visual Actions on a Robot using POMDPs. *Artificial Intelligence*, 174:704–725, 2010.
[23] 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. *RAS*, 54(11):933–943, 2006.
[24] G. Theocharous, K. Murphy, and L. P. Kaelbling. Representing Hierarchical POMDPs as DBNs for Multi-scale Robot Localization. In *International Conference on Robotics and Automation (ICRA)*, 2004.
[25] S. Thrun. Stanley: The Robot that Won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692, 2006.
[26] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms, 2009. `www.vlfeat.org`.