In *AAAI-11 Workshop on Automated Action Planning for Autonomous Mobile Robots* (PAMR 11), San Francisco, USA, August 7 2011.

# Visual Search and Multirobot Collaboration based on Hierarchical Planning

**Shiqi Zhang** and **Mohan Sridharan**
Stochastic Estimation and Autonomous Robotics Lab
Department of Computer Science
Texas Tech University
{s.zhang, mohan.sridharan}@ttu.edu

## Abstract

Mobile robots are increasingly being used in the real-world due to the availability of high-fidelity sensors and sophisticated information processing algorithms. A key challenge to the widespread deployment of robots is the ability to accurately sense the environment and collaborate towards a common objective. Probabilistic sequential decision-making methods can be used to address this challenge because they encapsulate the partial observability and non-determinism of robot domains. However, such formulations soon become intractable for domains with complex state spaces that require real-time operation. Our prior work enabled a mobile robot to use hierarchical partially observable Markov decision processes (POMDPs) to automatically tailor visual sensing and information processing to the task at hand (Zhang, Sridharan, & Li 2011). This paper introduces adaptive observation functions and policy re-weighting in a three-layered POMDP hierarchy to enable reliable and efficient visual processing in dynamic domains. In addition, each robot merges its beliefs with those communicated by teammates, to enable a team of robots to collaborate robustly. All algorithms are evaluated in simulated domains and on physical robots tasked with locating target objects in indoor environments.

## Introduction

In recent times, mobile robots have been used in many different application domains such as autonomous navigation, health care and disaster rescue due to the ready availability of high-fidelity sensors at moderate costs and the development of sophisticated sensory input processing algorithms (Thrun 2006; Hoey *et al.* 2010). Such real-world application domains are characterized by non-deterministic action outcomes, partial observability and dynamic changes. The sensory inputs are noisy and the corresponding processing algorithms extract information with different levels of uncertainty and computational complexity. Though partially observable Markov decision processes (POMDPs) (Kaelbling, Littman, & Cassandra 1998) elegantly encapsulate these characteristics of robots deployed in the real-world, such formulations are intractable because of the associated state space explosion and the high computational complexity of the corresponding solvers (Ross *et al.* 2008). Our previous work (Zhang, Sridharan, & Li 2011; Sridharan, Wyatt, & Dearden 2010) introduced a hierarchical decomposition of POMDPs that enabled a mobile robot to tailor visual sensing and information processing to the task at hand. However, this hierarchy was limited to visual processing on a single robot in non-cluttered (and mostly static) application domains.

This paper introduces adaptive observation functions and policy re-weighting in the POMDP hierarchy for reliable and efficient visual sensing and processing in dynamic domains. A probabilistic scheme is then used to enable each robot to merge beliefs acquired by sensing the environment with similar beliefs communicated by teammates, resulting in robust collaboration between a team of robots. All algorithms are evaluated on simulated and physical robots for the task of locating targets in dynamic indoor environments.



## Related Work

Classical planning methods compute a pipeline of visual operators for a high-level goal using deterministic action models whose preconditions and effects are propositions that need to be true or are made true by operator execution. However, in robot domains, the state is not directly observable and actions are unreliable.

In vision research, image interpretation has been modeled using MDPs and POMDPs. For instance, Li et al. (Li *et al.* 2003) use human-annotated images to determine the reward structure, explore the state space and compute value functions that are used for making the action choices. Similarly, active sensing has been used to decide sensor placement and information processing, e.g., using particle filters for estimating a multitarget probability density (Kreucher, Kastella, & Hero 2005). Sensor placements in spatial phenomena have also been modeled as Gaussian processes using submodular functions (Krause, Singh, & Guestrin 2008). However, many visual planning tasks are not submodular, and modeling probability densities using manual feedback over many trials is infeasible on robots.

Though a default POMDP formulation is well-suited for partially observable and non-deterministic domains, it is intractable for complex domains with large state spaces. Pineau and Thrun (2003) proposed a hierarchical POMDP approach where the top level action of a robot is a collection of simpler actions modeled as smaller POMDPs; planning occurs in a bottom-up manner, and execution proceeds in a top-down manner. Similar approaches have been used

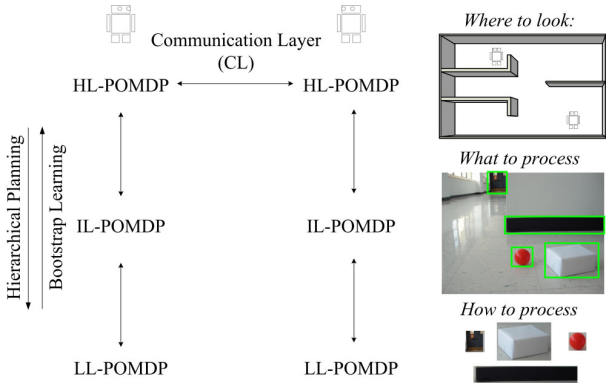



Figure 1: Overview

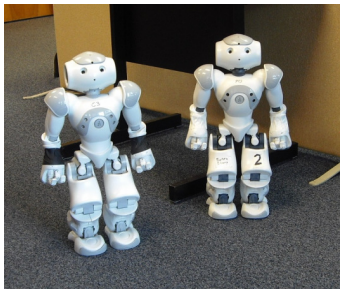


Figure 2: Robot test platforms

for robot navigation (Foka & Trahanias 2005) but a significant amount of data for the hierarchy and model creation is hand-coded. Recent work has focused on learning POMDP observation models (Atrash & Pineau 2010); using information maximization for POMDP-based visual search (Butko & Movellan 2008) and developing faster solvers (Ross *et al.* 2008). However, these methods are computationally expensive, or require considerable human input.

Several algorithms continue to be developed for multiagent and multirobot collaboration in a variety of domains (de Weerdt & Clement 2009; Panait & Luke 2005). Sophisticated algorithms have also been developed recently for using decentralized POMDPs (Dec-POMDPs) for multiagent collaboration (Kwak *et al.* 2010). However, the computational complexity of these formulations is more severe than the default POMDP formulations (Bernstein *et al.* 2002). Research has also shown that using complex communication strategies does not necessarily aid in better collaboration (Rybski *et al.* 2004). This paper hence proposes a probabilistic scheme that uses shared beliefs of teammates (obtained using the hierarchical POMDP) to enable robust multirobot collaboration despite unreliable communication.

## Hierarchical Planning and Collaboration

This section describes the proposed approach for reliable visual processing and robust multirobot collaboration. The algorithms are illustrated using the challenge task of a team of robots locating objects in dynamic indoor (office) domains. Experiments are performed on the humanoid and wheeled robots shown in Figure 2.

### Hierarchical POMDPs for Visual Processing

This section summarizes the three-layered POMDP hierarchy for visual sensing and processing on a robot. As shown in Figure 1, the high-level (HL) POMDP chooses a sequence of 3D scenes to process based on the task (*where to look?*). The intermediate-level (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?*). Each ROI uses a lower-level (LL) POMDP to compute the sequence of visual operators to be applied to address the specific task (*how to process?*). Our prior work used an instance of this hierarchy in non-cluttered (and mostly static) indoor domains (Zhang, Sridharan, & Li 2011). The HL-POMDP and the modifications introduced for dynamic domains are described here. The IL-POMDP and LL-POMDPs are summarized briefly—they have been used in earlier work for human-robot interaction in a simplistic tabletop scenario (Sridharan, Wyatt, & Dearden 2010).

Target objects can exist in different locations, and a robot has to move to analyze different scenes and locate the objects. Assume that the robot has a learned map of its world (Dissanayake, Newman, & Clark 2001) and has to locate a specific target. The robot represents the 3D area as a discrete 2D *occupancy grid*. Each grid cell stores the likelihood of target occurrence in that cell. The HL-POMDP then poses sensing as the task of maximizing information gain, i.e., reducing the entropy in the belief distribution over the grid map. For a grid with $N$ cells, the HL-POMDP tuple $\langle S^H, A^H, T^H, Z^H, O^H, R^H \rangle$ is defined as:

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

The robot maintains a *belief state*, a probability distribution over the state. The *entropy* of belief distribution $B_t$ is:

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

where $b^i$ is the $i^{th}$ entry of the belief distributed over the grid cells. An action's reward is the entropy reduction between belief state $B_{t-1}$ and the resultant belief state $B_t$:

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

As the belief distribution slowly converges from a uniform distribution to states likely to be target locations, the entropy reduces. Unlike our prior work (Zhang, Sridharan, & Li 2011), the observation function is defined adaptively based on the locations of obstacles and the expected performance of the lower levels of the hierarchy:

$$O(z_i = present, s_j, a_k) = \begin{cases} \beta & \textbf{if } isBlocked(s_j, a_k) \\ \eta \cdot e^{-\frac{\lambda\mu^2}{2\sigma^2}} & \textbf{otherwise} \end{cases} \quad (3)$$

where $p(z_i = present|s_j, a_k)$, the probability of finding the target in cell $i$ given that the target is in cell $j$ and focus is on cell $k$, is a Gaussian whose mean depends on the target's location, the cell being examined and the field of view: $\mu = f_\mu(s_j, a_k)$. The variance is based on the expected observations from the lower levels of the hierarchy: $\sigma^2 = f_{\sigma^2}(O, O^I|s_j, a_k)$. If there is any obstacle between cells $j$ and $k$, $\beta$ is a small probability that the target can still be observed. POMDP solvers take such a model and compute a *policy*: $\pi^H : B_t \mapsto a_{t+1}$ that maps beliefs to actions. This paper uses policy gradient methods to compute the HL policy that minimizes belief entropy), in the form of stochastic action choices, i.e., "weights" used to probabilistically choose actions (Buffet & Aberdeen 2009).

**Convolutional Policy** The HL-POMDP formulation becomes intractable for domains with large state spaces due to the high computational complexity of POMDP solvers. A convolutional policy kernel is hence learned by exploiting the (local) rotation and shift-invariance of visual search:

$$\bar{K}(s) = (\pi^H \otimes C_m^K)(s) = \int \pi^H(\tilde{s}) C_m^K(s - \tilde{s}) d\tilde{s}, \quad (4)$$

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

where $\bar{K}$ is the un-normalized kernel, $\pi^H$ is the HL-POMDP policy, $K$ is the normalized policy and $C_m^K$ is the convolution mask whose size decides the size of target kernel.

Consider Figure 3(a), where a $3 \times 3$ policy kernel is extracted from a $5 \times 5$ baseline policy. Each row of the baseline policy $\pi^H$ 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—left column of Figure 3(a). When a robot visits a grid cell, only the beliefs immediately around that grid cell change substantially. The *policy kernel* is hence learned by focusing on a local area, and setting all other weights to a much smaller value. For instance, the $3 \times 3$ policy kernel $\bar{K}$ is computed by convolving a $3 \times 3$ mask with these policy layers—middle column of Figure 3(a). In parallel, the number of accumulated weights for each action is counted and the matrix $W$ is used to obtain the normalized kernel $K$—right column of Figure 3(a).

The computed kernel does not assign action weights to the grid cells further 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:

$$W^B = \frac{\sum_{actions} \sum_{states} \pi^H - \sum_{actions} \sum \bar{K}}{N_{actions} \times N_{states} - \sum W} \quad (5)$$

where the default weight value is a function of the number of actions: $N_{actions}$ and the number of states: $N_{states}$.

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

$$\pi_C^H(s) = (K \otimes C_m^E)(s) = \int K(\tilde{s}) C_m^E(s - \tilde{s}) d\tilde{s} \quad (6)$$

where $\pi_C^H$ is the convolutional policy, $K$ is the policy kernel and $C_m^E$ is the convolution mask of the same size as the target map. Consider Figure 3(b), where the $3 \times 3$ kernel is convolved with a $7 \times 7$ mask to generate the policy for a $7 \times 7$ map. The desired policy is generated one layer at a time, by centering the kernel on the state represented by the layer—the current example has 49 layers. Since the kernel covers (at most) nine grid-cells, other cells are assigned the weight computed in Equation 5 and the policy is normalized.

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 the distance and the robot's speed:

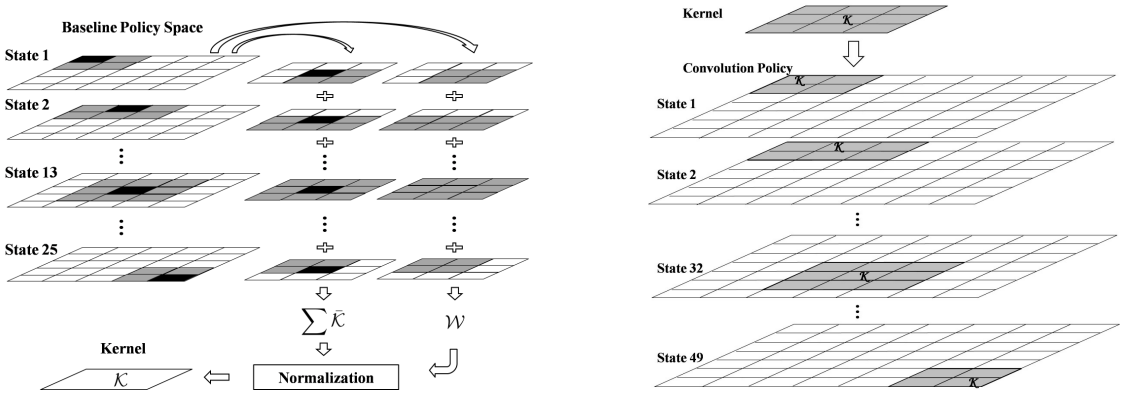
$$\hat{w}(i) = w(i) \frac{1}{1 + \frac{d_{A^*}(a_i, a_j)}{\text{speed}}} \quad (7)$$

where $d_{A^*}(a_i, a_j)$ is the distance between the current cell and the candidate cell, computed using the $A^*$ algorithm (Russell & Norvig 2003). The revised policy trades off likelihood of locating the target against the cost of traveling to that location. When the domain map changes, the revised distances change action weights for subsequent computations.

The overall operation of the POMDP hierarchy is as follows: the learned world map and models of the visual operators are used to generate the HL-POMDP model. The corresponding HL policy chooses a 3D scene to analyze next for the target object. The robot moves to this scene and captures images. Each salient region of interest (ROI) in an image is modeled as an LL-POMDP, where actions are information processing operators (e.g., detect color, object category). The corresponding LL policy provides the sequence of operators suitable to detect the target in a specific ROI. The LL policies of all image ROIs are used to create an IL-POMDP, and executing an action in the corresponding IL policy directs the robot's attention to a specific ROI. The result from the corresponding LL policy causes a belief update and action choice in the IL until the presence or absence of the target in the image is determined. The IL outcome causes an HL belief update and subsequent analysis of a grid cell until the target is found. The entire operation occurs reliably and efficiently due to the adaptive convolutional policies and automatic belief propagation. Furthermore, a high-level symbolic planner can be used to provide a sequence of goals (e.g., prioritized list of targets) that will be achieved using the proposed POMDP hierarchy.

## Multirobot Collaboration

Consider a team of $X$ robots that is tasked with locating $Y$ targets. Each robot maintains a separate belief vector (over the map) for each target, and tailors its sensing and visual processing using hierarchical POMDPs as described above. This section describes a probabilistic scheme for a team of robots to collaborate to locate all the targets.




(a) Kernel Extraction (b) Policy Extension

Figure 3: Convolutional Policy: (a) Extract 3×3 kernel from 5×5 baseline policy; (b) Extend 3×3 kernel to 7×7 policy.

For ease of explanation and without loss of generality, it is assumed that the targets are unique and that the observations of targets are independent of each other. Each robot stores:

$$\{B_i, f_i\},\ \forall i \in [1, |TL|] \tag{8}$$

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, the robot stores an action map $\mathscr{M}$, each of whose entries stores the number of times the robot has visited the corresponding grid cell:

$$\mathscr{M} = \langle m_1, \cdots, m_N \rangle \tag{9}$$

where $m_i$ is the count of the number of times grid-cell $i$ has been visited. The counts are updated after an action and the corresponding observation updates the appropriate belief vector. After the belief update, each robot broadcasts a package that includes its current belief vectors ($\forall i\, B_i$), discovery flags ($\forall i\, f_i$) and action map ($\mathscr{M}$).

Though the communicated estimates cannot be trusted completely, they provide useful information about map locations that the robot has not visited. Each robot therefore assigns a weight to own beliefs and communicated beliefs based on whether the robot generating this belief has recently observed the corresponding map region:

$$b_i^{j,own} = \frac{m_i^{j,own} \cdot b_i^{j,own} + m_i^{j,comm} \cdot b_i^{j,comm}}{m_i^{j,own} + m_i^{j,comm}} \tag{10}$$
$$\forall j \in [1, N], \quad \forall i \in [1, |TL|]$$

where $b_i^j$ is $j^{th}$ entry of the belief vector of target $i$, while $m_i^{j,own}$ and $m_i^{j,comm}$ are entries of the action maps of the robot and the teammate whose communicated belief is being merged. The discovery of each target is based on:

$$\mathscr{F} = \{f_i^{own} || f_i^{comm}; \forall i \in [1, |TL|]\} \tag{11}$$

where each target is assumed to be found when at least one robot has discovered it. Once a target has been discovered, a robot that requires a new target chooses an undiscovered object from the list ($TL$):

$$targetID = argmax_i \{\max_j B_i(j)\} \tag{12}$$

where the goal is to select the target whose location the robot is likely to discover with least effort, based on own and communicated beliefs. An additional heuristic cost based on distance of travel can be added, similar to Equation 7.

## Experimental Setup and Results

This section describes the results of experiments performed in simulation and on the robots of Figure 2. The goal is to evaluate the robot's ability to: (a) use adaptive convolutional policies (CC) and the POMDP hierarchy to achieve reliable and efficient visual sensing and processing in complex domains; and (b) probabilistically merge its beliefs with communicated beliefs of teammates to achieve robust collaboration. The execution of IL and LL policies are not described—the corresponding results are bundled into a single response for an HL action. The POMDP models are solved using a policy gradients algorithm and multirobot collaboration is evaluated in the context of searching for targets in dynamic indoor scenarios.

Since executing many trials on robots is infeasible, a realistic simulator was used to generate grid maps and randomly choose the initial positions of targets and robots. The simulated observations were based on the models learned by the robot. Unreliable wireless communication was simulated by varying the rate at which communicated packets were lost.

For robot experiments, objects were characterized by visual features, e.g., color and image gradients extracted using the *VLFeat* library (Vedaldi & Fulkerson 2009). Objects at known locations are used by the robot to collect image feature statistics, which are used to learn the LL observation functions and the feature representations for objects. The grid cells vary in size (1 − 3m) based on the field of view of cameras. Robots, boxes, chairs and other objects in the lab were used as targets.

Experiments evaluated three hypotheses: (I) the CC policy provides similar detection accuracy but is much more efficient than the non-convolutional (i.e., baseline) policy; (II) the CC policy results in higher reliability than an ad-hoc search strategy; and (III) the belief merging enables a team of robots to collaborate robustly.

### Simulation Experiments

In the simulation experiments, each data point is the average of 1000 trials. The figures also include robot trials wherever appropriate. In order to evaluate hypothesis I, a baseline policy was computed for a $5 \times 5$ map in a few hours—this is a one-time computation. A $5 \times 5$ policy kernel derived from

this policy was used to compute policies for larger maps. Figure 4(a) compares the CC policy against the baseline policy for a $7 \times 7$ map—the x-axis shows the number of times the policy was invoked, as a fraction of the number of states. A trial was successful if the location of the target was identified correctly. No statistically significant difference was observed in the performance of the CC and baseline policies.

To evaluate hypothesis II, the convolutional policy's performance was compared against an ad-hoc policy that defaults to random actions when there is no prior knowledge of target location. The results in Figure 4(b) used a $15 \times 15$ convolutional policy generated from a $5 \times 5$ kernel. The locations of the robot and the target were randomly selected for each trial. The data points were generated by terminating trials at different time steps—the grid cell with the largest belief value was then considered the target's location. Performance is scored as the weighted distance between the actual and detected locations of the target. Figure 4(b) shows that the CC policy significantly reduces the number of steps required to locate the target reliably. Over trials conducted on a range of grid maps, the hierarchical POMDPs with the adaptive CC policy result in an (average) accuracy of 96% in comparison to the 80% accuracy of ad-hoc strategy. The few errors correspond to situations where the target is at the edge of two grid cells—the estimated location is no more than one cell away from the actual location. When the domain map was changed dynamically, the robot adapted automatically, with an expected increase in the number of actions required to detect targets with the same accuracy.

To evaluate hypothesis III, all robots in a team were assumed to move at the same speed and the average distance moved by the robots in a team (in one trial) was used as a measure of the team's performance. Robots and targets were placed randomly in a grid map, with no more than one robot or target in each grid cell. When the belief in a grid cell exceeded 0.9, it was assumed to contain a target. Figure 4(c) shows the results for different combinations of robots and targets in a $15 \times 15$ grid map based on a real-world office scenario. The results show that the robots collaborate effectively to find the targets—similar results were obtained with grid maps of different sizes.

Figure 5(a) shows examples of the team's performance (for a specific number of robots and targets) as a function of prior knowledge. As expected, the robots are able to identify the targets faster when more information about target locations is available. Figure 5(b) shows results of trials, as a function of varying *communication success rate* (CSR), where robot teams were asked to locate two targets. Though a low likelihood of successful communication hurts the team's performance, it soon stabilizes and does not change significantly as CSR increases.

Figure 5(c) shows results corresponding to an experiment where two robots had to locate three targets in a $15 \times 15$ map, in comparison to an ad-hoc policy that assigns robots to targets heuristically and selects actions randomly. The proposed strategy takes a much smaller number of steps than the ad-hoc scheme to detect target objects with high accuracy, e.g., for an accuracy of 0.9, the average distance traveled using the proposed strategy is less than that traveled using the ad-hoc scheme by $\approx 88$ units, which corresponds to $\approx 13$ actions. Over extensive simulation experiments (and some robot trials) in different maps ($3 \times 3$ to $25 \times 25$), using the CC policy, hierarchical POMDPs and collaboration strategy enables a team of robots to collaborate robustly and identify desired targets reliably and efficiently.

### Robot Experiments

The algorithms were also evaluated on a team of robots in indoor scenarios—see Figure 1. The robots and target objects are randomly placed in the indoor domain that is discretized to form the grid map. When the robot moves to a specific grid cell, it processes salient regions (ROIs) of images of that scene to detect the target. The robots use UDP to communicate with each other. Two capabilities were evaluated: (a) each robot is able to locate objects reliably and efficiently; and (b) a team of robots is able to collaborate robustly.

The results in Figure 4(b) include the results of 30 trials on a robot in indoor corridors and offices that are mapped to a $15 \times 15$ grid with walls, doors and other obstacles. The CC policy performs much better than the ad-hoc policy on the robots—for a specific (high) accuracy level, the CC policy requires a much smaller number of steps than the ad-hoc policy. The target is recognized in the correct location during all experimental trials on the robot because the robot is able to acquire different views of the target.

The multirobot collaboration strategy was then compared against an ad-hoc strategy, a voting mechanism to distribute targets among robots using manually generated heuristics (Stone *et al.* 2006). Experiments consisted of 25 trials with 2-3 robots and 2-3 targets. The proposed strategy successfully identified all targets in all trials, in comparison to the 85% accuracy of the ad-hoc strategy, and required a smaller number of steps to detect targets.

## Conclusion

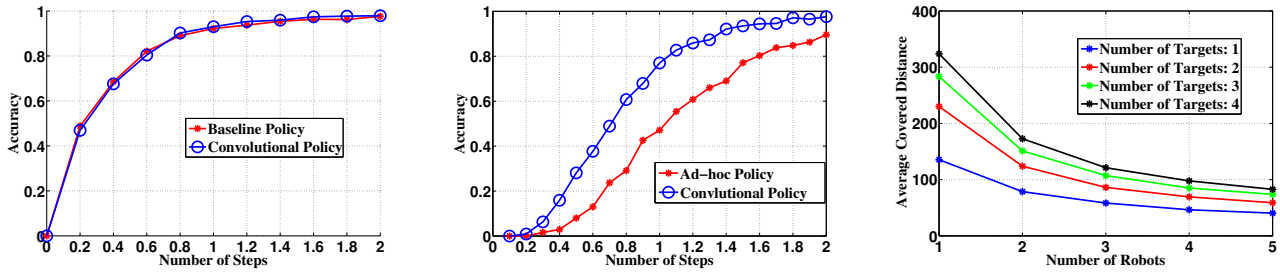
This paper described an approach for hierarchical visual processing and multirobot collaboration. The POMDP hierarchy enables a robot to sequence a subset of unreliable visual operators to automatically, reliably and efficiently tailor visual sensing and processing to the task at hand. The proposed multirobot collaboration scheme accounts for the unreliability of sensing and communication in the belief merging process, enabling a team of robots to collaborate robustly in dynamic domains. Future research will use learned models of the sensing and actuation capabilities of individual robots in the decision-making, and experiment with a larger number of robots and targets. The ultimate goal is to enable reliable, efficient and autonomous multirobot collaboration in real-world scenarios.

## References


Atrash, A., and Pineau, J. 2010. A Bayesian Method for Learning POMDP Observation Parameters for Robot Interaction Management Systems. In *The POMDP Practitioners Workshop*.

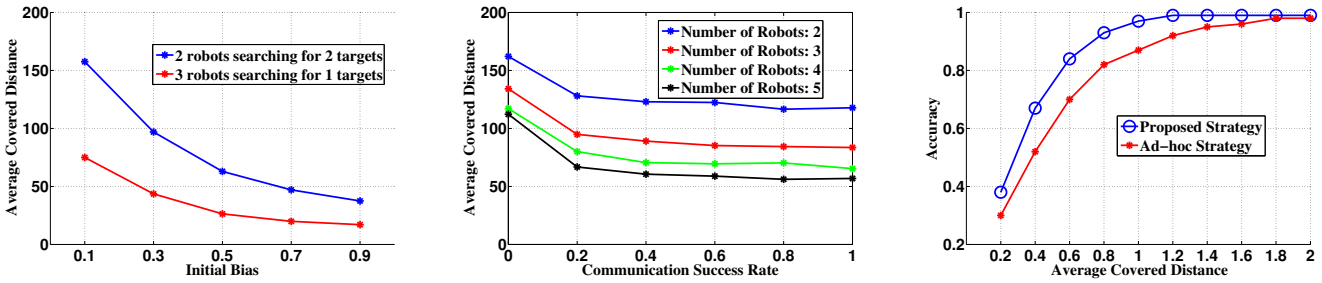
Bernstein, D. S.; Givan, R.; Immerman, N.; and Zilberstein, S. 2002. The Complexity of Decentralized Control of Markov Decision Processes. *Mathematics of Operations Research* 27(4).

Buffet, O., and Aberdeen, D. 2009. The Factored Policy-Gradient Planner. *Artificial Intelligence* 173(5-6):722–747.




(a) CC vs. Baseline. (b) CC vs. Ad-hoc. (c) Multirobot search.

Figure 4: (a) Detection accuracy with CC policies is similar to an expensive baseline policy; (b) CC policies performs better than a ad-hoc search strategy; (c) Belief merging and hierarchical POMDPs result in robust multirobot collaboration.




(a) Effect of Bias. (b) Effect of CSR. (c) Multirobot search.

Figure 5: (a) Performance improves if prior information is incorporated; (b) Performance is robust to dropped packages; and (c) Proposed strategy results in significantly better collaboration than ad-hoc strategy.


Butko, N. J., and Movellan, J. R. 2008. I-POMDP: An Infomax Model of Eye Movement. In *ICDL*.

de Weerdt, M., and Clement, B. 2009. Introduction to Planning in Multiagent Systems. *Multiagent and Grid Systems* 5(4):345–355.

Dissanayake, G.; Newman, P.; and Clark, S. 2001. A Solution to the Simultaneous Localization and Map Building Problem. *Transactions on Robotics and Automation* 17(3):229–241.

Foka, A. F., and Trahanias, P. E. 2005. Real-time Hierarchical POMDPs for Autonomous Robot Navigation. In *IJCAI Workshop on Reasoning with Uncertainty in Robotics*.

Hoey, J.; Poupart, P.; Bertoldi, A.; Craig, T.; Boutilier, C.; and Mihailidis, A. 2010. Automated Handwashing Assistance for Persons with Dementia using Video and a Partially Observable Markov Decision Process. *CVIU* 114(5):503–519.

Kaelbling, L.; Littman, M.; and Cassandra, A. 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence* 101:99–134.

Krause, A.; Singh, A.; and Guestrin, C. 2008. Near-optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *JMLR* 9:235–284.

Kreucher, C.; Kastella, K.; and Hero, A. 2005. Sensor Management using An Active Sensing Approach. *IEEE Transactions on Signal Processing* 85(3):607–624.

Kwak, J.; Yang, R.; Yin, Z.; Taylor, M.; and Tambe, M. 2010. Teamwork and Coordination under Model Uncertainty in DEC-POMDPs. In *AAAI Workshop on Interactive Decision Theory and Game Theory*.

Li, L.; Bulitko, V.; Greiner, R.; and Levner, I. 2003. Improving an Adaptive Image Interpretation System by Leveraging. In *Australian Conference on Intelligent Information Systems*.

Panait, L., and Luke, S. 2005. Cooperative Multi-Agent Learning: The State of the Art. *JAAMAS* 11(3):387–434.

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

Ross, S.; Pineau, J.; Paquet, S.; and Chaib-draa, B. 2008. Online Planning Algorithms for POMDPs. *JAIR* 32:663–704.

Russell, S. J., and Norvig, P. 2003. *Artificial Intelligence: A Modern Approach*. New Jersey: Prentice Hall.

Rybski, P. E.; Larson, A.; Veeraraghavan, H.; LaPoint, M.; and Gini, M. 2004. Communication strategies in Multi-Robot Search and Retrieval: Experiences with MinDART. In *DARS*.

Sridharan, M.; Wyatt, J.; and Dearden, R. 2010. Planning to See: A Hierarchical Aprroach to Planning Visual Actions on a Robot using POMDPs. *Artificial Intelligence* 174:704–725.

Stone, P.; Sridharan, M.; Stronger, D.; Kuhlmann, G.; Kohl, N.; Fidelman, P.; and Jong, N. K. 2006. From Pixels to Multi-Robot Decision-Making: A Study in Uncertainty. *Robotics and Autonomous Systems* 54(11):933–943.

Thrun, S. 2006. Stanley: The Robot that Won the DARPA Grand Challenge. *Journal of Field Robotics* 23(9):661–692.

Vedaldi, A., and Fulkerson, B. 2009. Vlfeat: An open and portable library of computer vision algorithms. `www.vlfeat.org`.

Zhang, S.; Sridharan, M.; and Li, X. 2011. To Look or Not to

Look: A Hierarchical Representation for Visual Planning on Mobile Robots. In *ICRA*.