The 4th International Conference on Field and Service Robotics, July 14–16, 2003

Planning under Uncertainty for Reliable Health Care Robotics

Nicholas Roy nickr@ri.cmu.edu
Geoffrey Gordon ggordon@cs.cmu.edu
Sebastian Thrun thrun@cs.cmu.edu

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We describe a mobile robot system, designed to assist residents of an retirement facility. This system is being developed to respond to an aging population and a predicted shortage of nursing professionals. In this paper, we discuss the task of finding and escorting people from place to place in the facility, a task containing uncertainty throughout the problem.

Planning algorithms that model uncertainty well such as Partially Observable Markov Decision Processes (POMDPs) do not scale tractably to most real world problems. We demonstrate an algorithm for representing real world POMDP problems compactly, which allows us to find good policies in reasonable amounts of time. We show that our algorithm is able to find moving people in close to optimal time, where the optimal policy would start with knowledge of the person's location.

1 Introduction

We describe a mobile robot system, designed to assist residents of an retirement facility. This system is being developed to respond to an aging population and a predicted shortage of nursing professionals. Previously, we have reported on work focused on the task of reminding people of events (e.g., appointments) and accompanying them to these events [1, 2]. In this paper, we discuss the task of finding and escorting people from place to place in the facility. This problem contains many common aspects of real world uncertainty: the state of the world is not completely known (the initial position of the person in the environment is unknown), the state changes (people are free to move around), and sensor noise can lead to perceptual errors. These sources of uncertainty, if unmodelled, can lead to sub-optimal behaviour on the part of the robot.

Unfortunately, the kind of planning that is required for reliable robot operation is difficult to approximate with simple heuristics for handling the uncertainty, so we must use a planning methodology that explicitly models the real-world uncertainty. One such model is the Partially Observable Markov Decision Process (POMDP), but conventional approaches to finding policies for POMDPs are often intractable for the size of problems we wish to address.

We will take advantage of dimensionality reduction techniques to find low-dimensional representations that can be planned for much more easily, by using structure inherent in many real world domains. For example, Principal Components Analysis (PCA) is well-suited to dimensionality reduction for data on or near a linear manifold in the higher-dimensional space. Unfortunately, POMDP belief manifolds are rarely linear; in particular, sparse beliefs are usually very non-linear. We therefore transform the data into a space where it does lie near a linear manifold; the algorithm which does so (while also correctly handling the transformed residual errors) is called Exponential Family PCA (E-PCA) [3, 4]. E-PCA will allow us to represent POMDPs with only a handful of dimensions, even for belief spaces with thousands of dimensions. We will demonstrate the use of this planning technique on the problem of how to find a person whose location is initially unknown.

2 Finding People

The problem we wish to solve is how to find people in a health care facility as quickly as possible. The robot is assumed to begin with a grid map of the environment, but no knowledge of where the person might be located, in which grid cell of the map. The robot can move about the environment to look for the person, and receives sensor information when the robot can and cannot see the person. Our implementation is based on a laser-range finder, but this work is independent of the particular sensing modality.

We assume a probabilistic state estimator that provides probability distributions over where people might be located. We will refer to this a distributions as a "belief" of the person's location. The belief is updated over time after each action and observation from the robot according to a

(a) Pearl (b) Pearl in Longwood (c) Pearl in Longwood

Figure 1: (a) Pearl, the Nursebot (b) & (c) Pearl interacting with residents of Longwood at Oakmont.

well-formed probabilistic rules [5]. The planning task can then be phrased as one of choosing the next action, based on the current belief, as depicted in figure 2. Not shown in this figure is the true state of the world, which is also not observable by the agent.

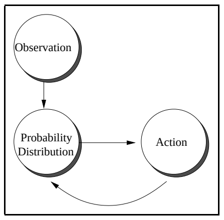


Figure 2: The execution process for finding people. The observation is generated according to an emission probability model conditioned on the current state, that is hidden from the controller. The controller *only* has access to the observation, not the true state generating the observation.

The Partially Observable Markov Decision Process is a decision-theoretic model for planning successfully with beliefs. The POMDP is solved by defining a "value function" over the space of beliefs, which assigns a value and action to each belief. By iteratively updating the value function appropriately, the value function can be made to converge to the greatest expected reward from each belief, and the action that will achieve that reward in expectation. The POMDP finds a policy that maximises the expected sum of future (possibly discounted) rewards of the agent executing the policy; for the problem of finding people, we can write a reward function for each possible configuration of the world such that the maximum reward is achieved for finding people fastest.

There are a large number of value function approaches [6, 7] that explicitly compute the expected reward of every belief. Such approaches produce complete policies (the optimal action for every belief), and can guarantee this optimality under a wide range of conditions. However, *finding* a value function this way is usually computationally intractable [6, 8].

Large POMDPs are generally very difficult to solve especially with standard value iteration techniques. Maintaining a full value function over the high-dimensional belief space entails finding the expected reward of every possible belief under the optimal policy. In reality, most POMDP policies generate only a small percentage of possible beliefs. For example, a mobile robot tracking a person is extremely unlikely to ever encounter a belief about the person's pose that resembles a checkerboard. If the execution of a POMDP is viewed as a trajectory inside the belief space, trajectories for most large, real world POMDPs lie on low-dimensional manifolds embedded in the belief space. So, POMDP algorithms that compute a value function over the full belief space do a lot of unnecessary work.

3 Dimensionality Reduction

In order to find the low-dimensional manifold for representing our belief space, we take advantage of dimensionality reduction techniques. One possible technique that we could consider is Principal Component Analysis[1] (PCA). We collect a data set of beliefs X, and use PCA to find a low-dimensional representation; so long as the collected data set is representative of the beliefs we will encounter during the execution of the people-finding plan, then we should be able to track the current belief on the low-dimensional manifold accurately.

PCA operates by finding a set of feature vectors $U = \{u_1, \ldots, u_n\}$ that minimise the loss function

$$L(U, V) = ||X - UV||^2 \tag{1}$$

[1]Also known as Singular Value Decomposition

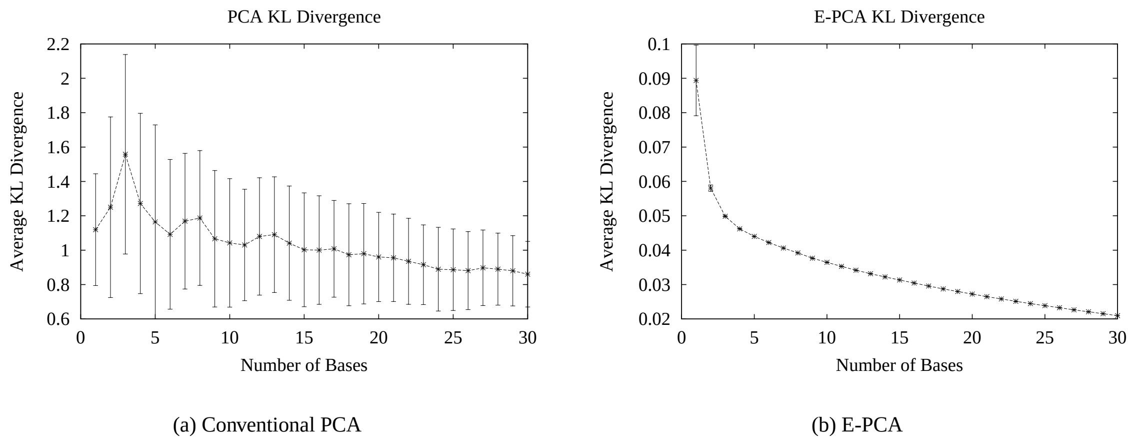


Figure 3: A comparison of the reconstruction quality of conventional PCA and E-PCA, using a probabilistic distance measure, Kullback-Leibler divergence, on a people-tracking data set. Notice that even with 30 bases, the PCA performs poorly and has very high variance in reconstruction quality. The E-PCA error falls rapidly initially, and the variance in error is low, indicating consistent performance across the entire data set. (Note the different scales on the Y axes.)

where X is the original data and V is the matrix of low-dimensional coordinates of X. This particular loss function assumes that the data lie near a linear manifold, and that displacements from this manifold are symmetric and have the same variance everywhere. (For example, i.i.d. Gaussian errors satisfy these requirements.) Unfortunately, probability distributions for POMDPs rarely form a linear subspace. In addition, squared error loss is inappropriate for modelling probability distributions: it does not enforce positive probability predictions.

We use exponential family PCA to address this problem. Other nonlinear dimensionality-reduction techniques [9, 10] could also work for this purpose, but would have different domains of applicability. Exponential family Principal Component Analysis [3] (E-PCA) varies from conventional PCA by adding a link function, in analogy to generalised linear models, and modifying the loss function appropriately. As long as we choose a link function that corresponds to an exponential family distribution log likelihood, and as long as the link and loss functions to match each other, there will exist efficient algorithms for finding U and V given X. By picking particular link functions (with their matching losses), we can reduce the model to an SVD.

In our case the entries of X are non-negative, and we wish to ensure accurate representation of low-probability events. Consequently, a link and loss function that correspond to the Poisson distribution are most appropriate[2]. The corresponding link function is

$$\bar{X} = f(UV) = \exp(UV) \qquad (2)$$

[2]Examples of other choices are the Exponential distribution, the multinomial, the Beta, etc. The Gaussian is also an Exponential family distribution, but a Gaussian link and loss function reduce to conventional PCA.

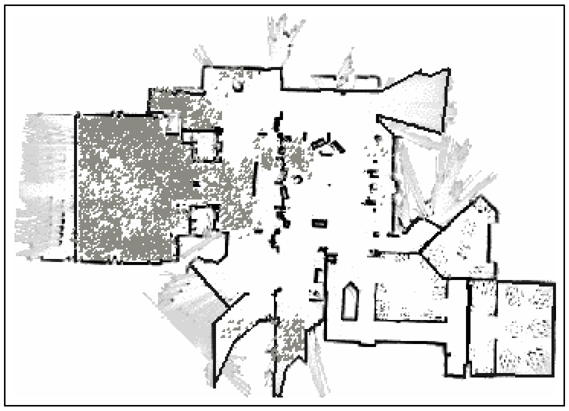
(taken component-wise) and its associated loss function is

$$L(U, V) = \exp(UV) - X \circ UV \qquad (3)$$

where the "matrix dot product" $A \circ B$ is the sum of products of corresponding elements. It is worth noting that using the Poisson loss for dimensionality reduction is related to Lee and Seung's non-negative matrix factorisation [11]. Gordon [12, 4] has a Newton's Method solution for computing U and V quickly.

In figure 3, we compared the error in the low-dimensional representations, for a sample set of 500 beliefs taken from the person tracking problem for the environment shown in figure 4. Figure 3(a) shows the average Kullback-Leibler divergence (a distance metric for probability distributions) between the high-dimensional belief set and the low-dimensional representation, using conventional PCA to find the low-dimensional representation. We see that the distance is large, does not improve quickly with more dimensions, and the representation quality is largely inconsistent, as denoted by the wide error bars. Figure 3(b) shows the same evaluation (average KL divergence) where E-PCA was used to find the low-dimensional representation. In this case, the error is small, improves quickly initially, and is consistent across the entire data set, in all ways outperforming conventional PCA.

We can also look at a sample representation to assess the quality of the representation. Figure 4 shows an example of the tracking process in progress. The true position of the person is unknown, and the robot instead maintains a probability distribution over possible poses of the person. The small grey dots show particles drawn from the original distribution. As the robot moves around the environment, sensor information is integrated into the distribution. The space of possible distributions is 1961-dimensional, for an


(a) Original distribution

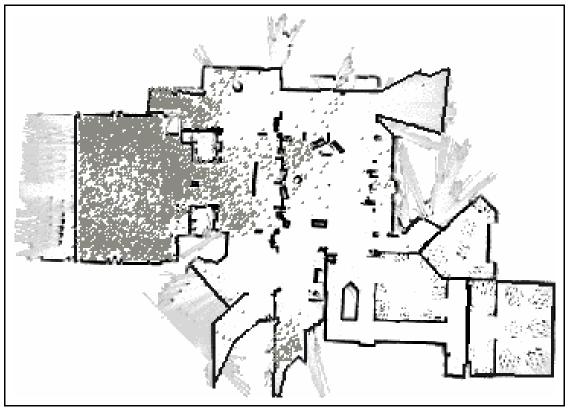

(b) Reconstruction

Figure 4: Examples of distributions in the Longwood at Oakmont retirement facility. The small grey dots show particles drawn from the original distribution; the higher the probability, the denser the particles. (a) An example distribution of potential positions of the person being searched for. This distribution is represented using 1961 dimensions. (b) The same distribution, reconstructed using only 6 dimensions. The true position of the person is not observable by the robot at a distance.

environment $53 \times 37m$ discretised into $1m$ grid cells, and 1 dimension for each grid cell. However, by taking advantage of the E-PCA decomposition we can generate a faithful representation of the space of *actual* distributions in only 6 dimensions. Figure 4(b) shows the original distribution projected to the low-dimensional space and then reconstructed. Although this is a lossy projection, the reconstruction is accurate for planning purposes. Remember that the task is not to reconstruct *only* the distribution shown in figure 4a, but to be able to represent *all* of the distributions that we expect to see as points in the 6-dimensional space.

4 Planning

Given the belief features acquired through E-PCA, it remains to compute the policy. Unfortunately, the non-linearity of the E-PCA projection prevents any guarantees of value function convexity over the low-dimensional space, which means that standard POMDP value iteration techniques cannot be used to find policies on the low-dimensional manifold directly. Instead, we approximate the low-dimensional space discretely, converting the POMDP into a belief space MDP. During execution, the action taken at each time step is taken from the discrete belief state that is closest to the current actual belief.

Our conversion algorithm from POMDP to MDP is a variant of the Augmented MDP, or Coastal Navigation algorithm [13], using belief features instead of entropy. We can compute the model reward function $\mathcal{R}(s_i)$ easily from the reconstructed beliefs, using $\mathcal{R}(b) = b \cdot \mathcal{R}(s)$. To learn the transition function $p(b_i|a, b_j)$, we can sample states from the reconstructed beliefs, sample observations from those states, and incorporate those observations to produce new belief states. Table 1 outlines the steps of this algorithm.

1. Collect sample beliefs
2. Use E-PCA to generate low-dimensional belief features
3. Convert low-dimensional space into discrete space $\mathcal{S}$
4. Learn belief transition probabilities $\mathcal{T}(s_i, a, s_j)$, and reward function $\mathcal{R}(s_i)$.
5. Perform value iteration on new model, using states $\mathcal{S}$, transition probabilities $\mathcal{T}$ and $\mathcal{R}$.

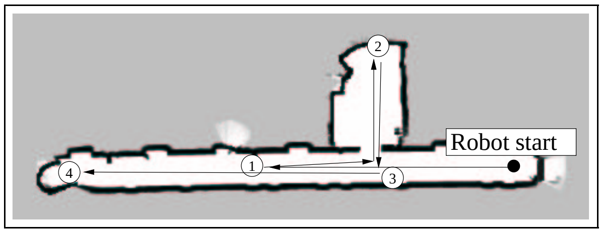
Table 1: Algorithm for planning in low-dimensional belief space.

The state space can be discretised in a number of ways, such as laying a grid over the belief features or using distance to the closest training beliefs to divide feature space into Voronoi regions. Thrun [14] has proposed nearest-neighbor discretisation in high-dimensional belief space; we propose instead to use nearest-neighbour in a low-dimensional feature space, where neighbors should be more closely related.

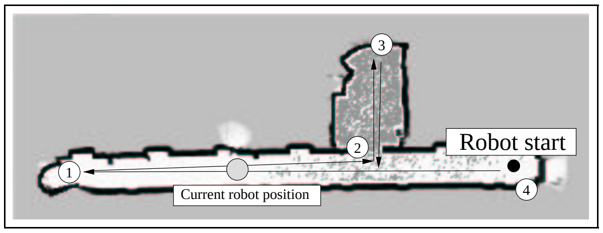
In order to find a good policy, we must be sure to discretise carefully. In some regions of the low-dimensional manifold, beliefs that are close together we can cluster into the same, large discrete cell without hurting performance. In other regions of the belief space, the cells must be much smaller, in order to distinguish different beliefs that require different actions. This leads to a variable resolution representation of the low-dimensional manifold.

We typically do not have enough belief samples initially to determine the full discretisation across the entire space; in places the discretisation will be insufficiently fine. We compensate by periodically re-evaluating the model at each grid cell, and splitting the grid-cell into smaller discrete cells where the model disagrees with some statistics of the real world. A number of different statistics have been suggested for testing the model against data from the real world [15], such as reduction in reward variance, or value function disagreement. We have opted instead for a simpler criterion of transition probability disagreement, although one improvement we are exploring is to use the Kolmogorov-Smirnov criterion for reducing expected reward disagreement [16].

5 Performance



(a) Optimal trajectory



(b) Sub-optimal trajectory

Figure 5: Example trajectory. (a) Even for this very simple environment, in order to maximise the likelihood of finding the person, the trajectory is relatively complicated. (b) The more obvious, sub-optimal trajectory allows some probability mass to "leak" into already-explored regions.

Figure 5(a) shows an example trajectory for a simple environment[3]. Even for this very simple problem, the trajectory is relatively complicated. The robot starts at the far end of the corridor, with the person's position completely unknown (the initial belief is uniform over the entire space). The robot travels past the open door on the right, part way down the corridor, returns to explore the room, and then finishes the corridor. This trajectory ensures that by the time the robot is finished exploring the room, the person must either have been found, or be at the far end of the corridor – there is no possibility for the person to escape into already-explored sections of the environment. This is an example of the kind of planning we hope to see – our planner has found a strategy that is not obvious, nor easy to capture using simple heuristics. Figure 5(b) shows a more obvious but sub-optimal trajectory in mid-execution. Notice the probability mass that appears in the already-explored region near the robot start location, causing the robot eventually to retrace its steps. The optimal strategy in figure 5(a) explicitly avoids this problem.

[3]For this environment, the original space was $47m \times 17m$ with a $0.2m$ resolution, for 20,230 grid cells, reduced to 6 dimensions.

Figure 6 shows a quantitative comparison of our technique and other possible heuristics. The horizontal line is the baseline, "True MDP" situation where the position of the person is always known correctly, that is, there is no hidden state. This algorithm is essentially cheating, but serves as a useful lower bound in that the robot find the person as quickly as possible every iteration. The "Closest" heuristic takes the robot to the nearest grid cell where the person might be. The "Densest" heuristic takes the robot to the location where the most particles are visible. The "MDP" heuristic takes the robot to the maximum-likelihood location (the single grid cell with the most particles). The "E-PCA 72" and "E-PCA 260" is a comparison of the E-PCA plans before state splitting (with 72 low-dimensional belief states) and after iterative refinement of the manifold (to 260 low-dimensional belief states). The "E-PCA 260" is clearly the best performing algorithm, able to find the person almost as quickly as the fully-observable planner.

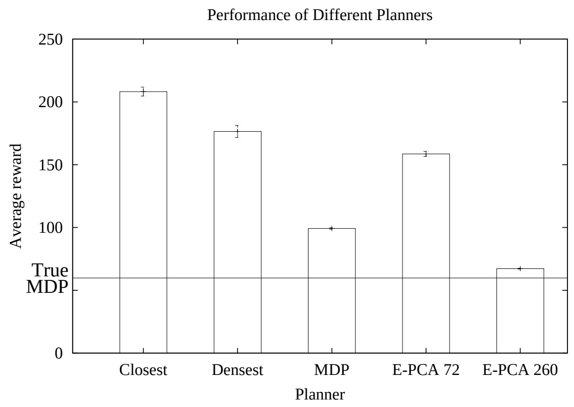


Figure 6: A comparison of different planning methods, including some simple heuristic planners. The"True MDP" method is the lower-bound "cheating" solution which assumes that the true position of the robot is always known. The optimal method under uncertainty is the "E-PCA-260" method, which also learns the optimal state decomposition.

6 Related Work

There have been a number of recent advances in solving large POMDPs. Poupart & Boutilier [17] make use of a similar dimensionality reduction technique, however, their representation requires a linear combination of bases to represent arbitrary data, which is a strong limitation on the compression they can achieve. (Figure 3(a) demonstrates the limitation of linear representations.) Pineau et al. [18]

have had success in finding approximate value functions quickly, but again their approach has not scaled to the size of the problems discussed in this paper.

Policy search algorithms [8, 19] have addressed some large problems. We suggest that a large part of the success of policy search is due to the fact that it focuses computation on relevant belief states. A disadvantage of policy search, however, is that can be data-inefficient across problems: many policy search techniques have trouble reusing sample trajectories generated from old policies. Our approach focuses computation on relevant belief states, but also allows us to use all relevant training data to estimate the effect of any policy.

Related research has developed heuristics which reduce the belief space representation. In particular, entropy-based representations for heuristic control [20] and full value-function planning [13] have been tried with some success. However, these approaches make strong assumptions about the kind of uncertainties that a POMDP generates. By performing principled dimensionality reduction of the belief space, our technique should be applicable to a wider range of problems.

7 Conclusion

We have demonstrated a system for finding and tracking people in the health care setting. The problem of finding people is computationally difficult in many environments, because of the high degree of uncertainty. Planners that do not reason intelligently about this uncertainty can take arbitrarily long to perform such real world tasks. The Partially Observable Markov Decision process is a planner that can reason about uncertainty, but is typically held not to scale to large problems.

We have shown that by taking advantage of dimensionality reduction techniques, we can represent POMDP problems compactly, and therefore generate good plans. We used a variant of PCA called Exponential family PCA (E-PCA) to find a low-dimensional manifold one which typical beliefs lie, and compute a value function over that manifold using a function approximator. We have also shown that naive function approximation is not sufficient for finding good plans. Our experimental results indicate that the optimal plan can be sensitive to small changes to the function approximation in different regions of the low-dimensional manifold. By appropriate use of statistical tests, we are able to find good variable resolution representations for the value function that lead to good policies.

References

[1] Nicholas Roy, Joelle Pineau, and Sebastian Thrun. Spoken dialog management for robots. In *The Proceedings of the Association for Computational Linguistics*, 2000.

[2] Michael Montemerlo, Joelle Pineau, Nicholas Roy, Sebastian Thrun, and Vandana Verma. Experiences with a mobile robotic guide for the elderly. In *Proceedings of the National Conference of Artificial Intelligence (AAAI 02)*, Edmonton,AB, July 2002.

[3] M. Collins, S. Dasgupta, and R. E. Schapire. A generalization of principal components analysis to the exponential family. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, Cambridge, MA, 2002. MIT Press.

[4] Nicholas Roy and Geoffrey Gordon. Exponential family PCA for POMDPs. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, 2003.

[5] Matthew Rosencrantz, Geoffrey Gordon, and Sebastian Thrun. Locating moving entities in indoor environments with teams of mobile robots. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.

[6] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[7] Milos Hauskrecht. Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 13:33–94, 2000.

[8] Andrew Ng and Michael Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2000.

[9] Sam Roweis and Lawrence Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[10] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.

[11] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[12] Geoffrey Gordon. Generalized^2 linear^2 models. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.

[13] Nicholas Roy and Sebastian Thrun. Coastal navigation with mobile robots. In *Advances in Neural Processing Systems 12*, pages 1043–1049, 1999.

[14] Sebastian Thrun. Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems 12*, 1999.

[15] Remi Munos and Andrew Moore. Variable resolution discretization for high-accuracy solutions of optimal control problems. In *International Joint Conference on Artificial Intelligence (IJCAI 99)*, 1999.

[16] Andrew McCallum. Instance-based utile distinctions for reinforcement learning. In *The Proceedings of the Twelfth International Machine Learning Conference (ICML)*, Lake Tahoe, CA, 1995.

[17] Pascal Poupart and Craig Boutilier. Value-directed compression of POMDPs. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, 2003.

[18] Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2003)*, Acapulco, Mexico, August 2003.

[19] J. Andrew Bagnell and Jeff Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the International Conference on Robotics and Automation*, 2001.

[20] Anthony R. Cassandra, Leslie Pack Kaelbling, and James A. Kurien. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. In *Proceedings of the IEEE/RSJ Interational Conference on Intelligent Robotic Systems (IROS)*, 1996.