# Solving POMDPs with Continuous or Large Discrete Observation Spaces

**Jesse Hoey**
Department of Computer Science
University of Toronto
Toronto, ON, M5S 3H5
jhoey@cs.toronto.edu

**Pascal Poupart**
School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
ppoupart@cs.uwaterloo.ca

## Abstract

We describe methods to solve partially observable Markov decision processes (POMDPs) with continuous or large discrete observation spaces. Realistic problems often have rich observation spaces, posing significant problems for standard POMDP algorithms that require explicit enumeration of the observations. This problem is usually approached by imposing an *a priori* discretisation on the observation space, which can be sub-optimal for the decision making task. However, since only those observations that would change the policy need to be distinguished, the decision problem itself *induces* a lossless partitioning of the observation space. This paper demonstrates how to find this partition while computing a policy, and how the resulting discretisation of the observation space reveals the relevant features of the application domain. The algorithms are demonstrated on a toy example and on a realistic assisted living task.

## 1 Introduction

Partially observable Markov decision processes (POMDPs) [1] provide a rich framework for planning under uncertainty. In particular, POMDPs can be used to robustly optimize the course of action of complex systems despite incomplete state information due to poor or noisy sensors. For instance, in mobile robotics [15], spoken-dialog systems [21; 25] and vision-based systems [7], POMDPs can be used to optimize controllers that rely on the partial and noisy information provided by various sensors such as sonars, laser-range finders, video cameras and microphones. Unfortunately, to date, the use of POMDPs in such real-world systems has been limited by the lack of scalable algorithms capable of processing rich and continuous sensor observations.

While *model-free* approaches such as neuro-dynamic programming [3], Monte Carlo sampling [23] and stochastic gradient descent [13; 2; 16] can tackle POMDPs with arbitrary observation spaces, they tend to require a large amount of simulation or a priori knowledge to restrict the space of policies (which reduces the need for simulation). Significant progress has also been made in developing approximate scalable algorithms for *model-based* POMDPs with large state spaces [20; 19] and complex policy spaces [17; 19; 24], however model-based algorithms cannot tackle problems with continuous nor large discrete observation spaces.

In this paper we study and propose new algorithms for model-based POMDPs with continuous or large discrete observation spaces. We first demonstrate that the complexity of observation spaces can often be significantly reduced without affecting decision quality. Intuitively, observations provide information to the decision maker for choosing a future course of action. When the same course of action is chosen for two different observations, these observations are indistinguishable from a decision making point of view, and can therefore be aggregated. Hence, when a policy is composed of a small set of *conditional plans* (conditional on the observations), it is possible to partition the observation space in a small number of regions corresponding to the relevant features of the observation space for decision making. Many systems tackle the feature detection problem separately from the decision making problem, first building a set of features, then computing a policy based on those observation features. In this paper, we demonstrate how the decision problem can be used to automatically define a set of relevant features that are sufficient to find an optimal policy.

The paper first provides some background on POMDPs in Sect. 2, followed by a discussion of the partitioning of the observation space in Sect. 3. Sects. 4 and 5 discuss methods for the one-dimensional and multi-dimensional cases, respectively. Sect. 6 reports experiments with an assisted living task.

## 2 Partially Observable MDPs

Formally, a POMDP is specified by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{Z}, T, Z, R, \gamma, h \rangle$ which consists of a set $\mathcal{S}$ of states $s$ capturing the relevant features of the world, a set $\mathcal{A}$ of actions $a$ corresponding to the possible control decisions, a set $\mathcal{Z}$ of observations corresponding to sensor readings, a transition function $T(s, a, s') = \Pr(s'|s, a)$ encoding the stochastic dynamics of the world, an observation function $Z(a, s', z) = \Pr(z|a, s')$ indicating how sensor readings relate to physical states of the world, a reward function $R(s, a)$ encoding the objectives of the system, a discount factor $\gamma$ (between 0 and 1) and a planning horizon $h$ (assumed to be infinite in this paper). We assume that states and actions are discrete, but observations can be continuous or discrete.

While it is common for observations to be continuous because sensors often provide continuous readings, states are often abstract, unobservable quantities. In the case of user modeling problems and event recognition problems, sates are often discrete. For continuous observations, $Z(a, s', z) = pdf(z|a, s')$ is a probability density function. Since states are not directly observable, the decision maker's belief about the current state is represented by a probability distribution $b(s) = \Pr(s)$ called *belief state*. After executing $a$ and observing $z$, the belief state $b$ is revised according to Bayes' theorem: $b^a_z(s') \propto \sum_s b(s)T(s, a, s')Z(a, s', z)$.

To illustrate the concepts we will present, we use the classic Tiger problem [5], in which the decision maker is faced with two doors. A small reward lies behind one door, but a large cost (a tiger) lies behind the other. The decision maker can either open a door, or listen for the tiger. Listening gives the decision maker information from which she can infer the location of the tiger. In the original, discrete, version of this problem, two possible observations result from listening (*left* or *right*), and these observations correspond with the true location of the tiger with probability $0.85$. In the continuous version we will discuss here, the decision maker has access to the original microphone array measurement that (noisily) locates the tiger in the continuous horizontal dimension (from the far left to the far right).

At each time step, the decision maker picks an action to execute based on the information gathered in past actions and observations. We can represent the decision maker's possible strategies by a set $CP$ of conditional plans $cp$ which correspond to decision trees. Fig. 1(a) shows the decision tree of a k-step conditional plan for the Tiger POMDP with discrete observations. Nodes are labeled by actions and edges are labeled by observations. The execution of a conditional plan starts at the root, performing the actions of the nodes traversed and following the edges labeled with the observations received from the environment. For example, the conditional plan in Fig. 1(a), will lead to opening a door if two successive observations confirm the same tiger location. We can define recursively a $k$-step conditional plan $cp_k = \langle a, os_{k-1}\rangle$ as a tuple consisting of an action $a$ with an observation strategy $os_{k-1} : \mathcal{Z} \rightarrow \mathcal{CP}_{k-1}$ that maps observations to $(k-1)$-step conditional plans. For POMDPs with continuous observations, conditional plans $cp$ are decision trees with infinitely many branches and observation strategies $os$ are continuous functions. The value function $\alpha_{cp}(b)$ of a conditional plan $cp$ is the expected sum of discounted rewards that will be earned when starting in belief state $b$. This value function is often called an $\alpha$-*vector* since it is linear with respect to the belief space and therefore parameterized by a vector $\alpha_{cp}(s)$ that has one component per state (i.e., $\alpha_{cp}(b) = \sum_s b(s)\alpha_{cp}(s)$). Fig. 1(b) shows the 5 $\alpha$-vectors corresponding to the conditional plans shown in Fig. 1(a). The $\alpha$-vectors corresponding to the conditional plans starting with an open door action ($\alpha_4$ and $\alpha_5$) have high value at one extreme of the belief space (when the certainty about the tiger location is high), but very low value at the other extreme.

A collection of conditional plans forms a policy $\pi$. The value function $V^\pi$ of a policy $\pi$ is the best value achieved by any of its conditional plans (i.e., $V^\pi(b) = \max_{cp\in\pi} \alpha_{cp}(b)$).

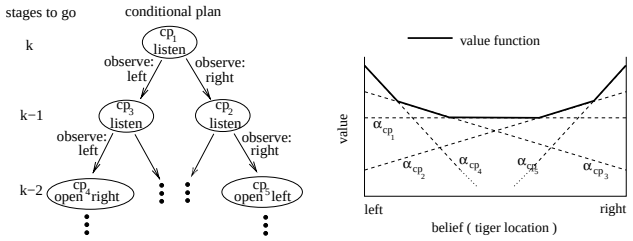



Figure 1: a) Tree representation of a three-step conditional plan for the simple tiger problem, starting with a uniform belief about the tiger location. b) Corresponding value function of composed of 5 $\alpha$-vectors

Fig. 1(b) shows the value function of a policy for the Tiger problem, composed of 5 conditional plans, each of which is highest for some part of the belief space. The goal is to find an optimal policy $\pi^*$ with the highest value (i.e., $V^{\pi^*}(b) \geq V^\pi(b)\ \forall\pi$). The optimal value function $V^*(b)$ for an infinite planning horizon can be computed by *value iteration* which computes successive approximations $V^k(b)$ by *dynamic programming*

$$V^{k+1}(b) = \max_a R^a(b) + \gamma \sum_z \Pr(z|b, a)V^k(b^a_z)$$

where $R^a(b) = \sum_s b(s)R(s, a)$, $\Pr(z|b, a) = \sum_{s,s'} b(s)\Pr(s'|s, a)\Pr(z|a, s')$ and $b^a_z$ is the revised belief state after executing $a$ and observing $z$. When $V^k$ is formed by the set of $\alpha_{cp}$, a new set of $\alpha_{cp'}$ for $V^{k+1}$ can be constructed by *point-based dynamic programming backups*. A point-based backup computes the best conditional plan $cp'^* = \langle a^*, os^*\rangle$ (and corresponding $\alpha$-vector) from a set of conditional plans $cp$ for a given belief state $b$:

$$\alpha_{\langle a^*, os^*\rangle}(b) = R^{a^*}(b) + \gamma \sum_z \Pr(z|b, a^*)\alpha_{os^*(z)}(b^{a^*}_z) \quad (1)$$

s.t.
$$os^*(z) = argmax_{cp}\ \alpha_{cp}(b^{a^*}_z) \quad (2)$$
$$a^* = argmax_a\ R^a(b) + \gamma \textstyle\sum_z \Pr(z|b, a)\alpha_{os^*(z)}(b^a_z) \quad (3)$$

In practice, we cannot perform such point-based backups for every belief state since the belief space is continuous. However, as noted by Smallwood and Sondik [22], finite-horizon optimal value functions are composed by a finite number of $\alpha$-vectors, which means that we only need to compute a finite number of point-based backups, one per $\alpha$-vector to be constructed. Exact algorithms such as Linear Support [6] and Witness [12] perform this finite set of point-based backups at carefully chosen belief points. Alternatively, approximate algorithms such as Point-Based Value Iteration [27; 17; 24] heuristically sample a set of belief points at which they perform point-based backups.

## 3 Policy-directed Observation Aggregation

In a decision process, observations provide information to the decision maker for deciding the future course of action. When the observation space is rich (i.e., continuous observations or many discrete observations), the decision maker can devise

rich policies with a different course of action for each possible observation. However, for many POMDPs, there often exists good policies that are quite simple. These policies tend to select the same course of action for many different observations that share similar features. For POMDPs with continuous observations, this will allow us to implicitly discretise the observation space without introducing any error. From the point of view of the application domain, this also gives us insights regarding the relevant features of the observation space. In this section we discuss how simple policies allow us to aggregate many observations, effectively reducing the complexity of the observation space.

Recall that a conditional plan $cp = \langle a, os \rangle$ is a tuple consisting of an action $a$ and an observation strategy $os$. The observation strategy $os(z) = cp'$ indicates for each observation $z$ the conditional plan $cp'$ encoding the future course of action. Intuitively, all the observations that select the same conditional plan are indistinguishable and can be aggregated. We can therefore view observation strategies as partitioning the observation space into regions mapped to the same conditional plan.[1] In the continuous observation tiger problem, for example, as long as a sound is heard coming "from the left", the best choice of action may be to open the right door. Although the precise location of the sound here is not important, the decision boundary is (e.g. how far right can the sound be heard before the decision maker would listen again).

In each point-based backup, we compute an observation strategy $os$ which partitions the observation space into regions $\mathcal{Z}_{cp}$ that select the same conditional plan $cp$. Let's examine how these regions arise. Recall from Eq. 2 that for each observation $z$, the conditional plan selected is the one that maximizes $\alpha_{cp}(b^a_z)$. Hence, we define $\mathcal{Z}_{cp^*} = \{z | cp^* = argmax_{cp} \alpha_{cp}(b^a_z)\}$ to be the set of observations for which $cp^*$ is the best conditional plan to execute in $b^a_z$.

For each region $\mathcal{Z}_{cp}$, we can compute the aggregate probability $\Pr(\mathcal{Z}_{cp}|a, s')$ that any observation $z \in \mathcal{Z}_{cp}$ will be made if action $a$ is taken and state $s'$ is reached by integrating $pdf(z|a, s')$ over region $Z_{cp}$ (i.e., $\Pr(\mathcal{Z}_{cp}|a, s') = \int_{z \in \mathcal{Z}_{cp}} pdf(z|a, s')dz$.[2] The aggregate probabilities can be used to perform point-based dynamic programming backups

$$\alpha_{\langle a^*, os^* \rangle}(b) = R^{a^*}(b) + \gamma \sum_{cp} \Pr(\mathcal{Z}_{cp}|b, a^*) \alpha_{cp}(b^{a^*}_{\mathcal{Z}_{cp}}) \quad (4)$$

with $\Pr(\mathcal{Z}_{cp}|b, a^*) = \sum_{s,s'} b(s) \Pr(s'|s, a^*) \Pr(\mathcal{Z}_{cp}|a^*, s')$ and $b^{a^*}_{\mathcal{Z}_{cp}}(s') \propto \sum_s b(s) \Pr(s'|s, a^*) \Pr(\mathcal{Z}_{cp}|a^*, s')$. This is equivalent to Eq. 1, except that we have replaced the sum over observations $z$ with a sum over regions $Z_{cp}$, in each of which a particular conditional plan is dominant.

## 4 One-Dimensional Observation Space

We now discuss how to find the implicit discretization of the observation space induced by a set of conditional plans (or $\alpha$-vectors) when the observation space is one-dimensional continuous. For this special case, the regions, $Z_{cp}$, over which observations can be aggregated are segments of a line corresponding to a range of observations for which the same conditional plan is optimal. Segment boundaries are observations for which there are two (or more) conditional plans yielding the highest $\alpha_{cp}(b^a_z)$. To make clear that $z$ is the only variable here, define $\beta^{b,a}_{cp}(z)$ to be a function in $z$ that corresponds to $\alpha_{cp}(b^a_z)$ where $b$ and $a$ are fixed (i.e., $\beta^{b,a}_{cp}(z) \equiv \alpha_{cp}(b^a_z)$). We can find these boundaries by solving $\beta^{b,a}_{cp_i}(z) - \beta^{b,a}_{cp_j}(z) = 0$ for every pair $cp_i$, $cp_j$ of conditional plans.[3] Analytically solving this equation will be difficult in general, and is not possible for observation functions in the exponential family (e.g. Gaussians). However, efficient numerical solutions can be used for many well-behaved functions. We used the Mathematica function `IntervalRoots`, that finds all the roots of an arbitrary one-dimensional function in a given interval by interval bisection combined with gradient-based methods. Once all potential regions are identified, the aggregate probabilities $P(\mathcal{Z}_{cp}|b, a)$ can be computed by exact integration (or Monte Carlo approximation) for each conditional plan $cp$.

Consider again the continuous Tiger problem introduced in Sect. 2. We now illustrate how to find the observation regions induced by a set of conditional plans and how to use them in a point-based backup. Suppose that the doors are located at $z = 1$ and $z = -1$, and the decision maker is at $z = 0$. Due to a lack of accuracy, the binary microphone array reports the tiger's location corrupted by some zero-mean, normally distributed noise of standard deviation $\sigma = 0.965$. Listening costs 1, while meeting the tiger costs 100, but opening the correct door is rewarded with 10. The discount is $\gamma = 0.75$.

Suppose we have 3 conditional plans with corresponding $\alpha$-vectors shown in Fig. 2(b) and we would like to perform a point-based backup. When considering belief state $b(tiger\_location = left) = 0.85$ and action $a = listen$, Fig. 2(a) shows the $\beta$ functions of each conditional plan. Since the observation function is Gaussian, the $\beta$-function is a linear combination of Gaussian distributions. By finding the roots of $\beta^{b,a}_{cp_1}(z) - \beta^{b,a}_{cp_2}(z)$ and $\beta^{b,a}_{cp_1}(z) - \beta^{b,a}_{cp_3}(z)$, we obtain the boundaries $z = 0.28$ and $z = 1.33$ delimiting the observation regions $\mathcal{Z}_{cp_1}$, $\mathcal{Z}_{cp_2}$ and $\mathcal{Z}_{cp_3}$.

We can thus form a discrete observation function $\Pr(\mathcal{Z}_{cp}|s, a)$ by integrating the original Gaussian observation distributions over each region. We analytically integrate each Gaussian over each region using the complementary error function erfc.[4] Fig. 2(c) shows the two Gaussian observation distributions, and the aggregate observation probabilities for each region. Using Eq. 4, we can then compute the value of the conditional plan $cp' = \langle a, \sigma \rangle$ where $a = listen$ and $\sigma(z) = cp_i$ if $z \in \mathcal{Z}_{cp_i}$.

In contrast, the original discrete version of the tiger problem partitions the observation space in two halves at $z = 0$, resulting in a discrete, binary observation function with $P(observe = right|tiger\_location = right) = \frac{1}{2}\text{erfc}(\frac{-1}{\sqrt{2}\sigma})$. A dynamic partition induced by the current set of conditional

[1] Note that the observations that are mapped to the same conditional plan may not form a contiguous region though.

[2] If $Z_{cp}$ is not a contiguous region, then several integrals must be computed, one for each contiguous sub-region.

[3] This method may find more boundaries than are necessary, for a third $\beta_{cp_k}$ may have higher value than both $\beta_{cp_i}$ and $\beta_{cp_j}$ at their intersection points.

[4] $\text{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt$.

plans has the advantage that no information is lost. Fig. 2(d) compares the value of the policies obtained with our dynamic discretization scheme and the naive binary discretization as we vary the variance $\sigma^2$. The dynamic discretization outperforms the fixed binary discretization for all variances. The solutions are the same for almost perfectly observable cases ($\sigma \leq 0.1$) and approach one another for almost unobservable cases ($\sigma \to \infty$).

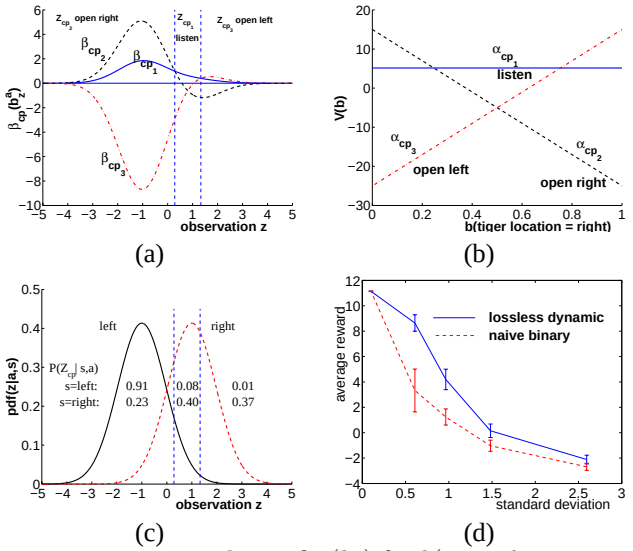



Figure 2: Tiger example. a) $\beta_{cp}(b_z^a)$ for $b(tiger\_location = left) = 0.85$ and $a = listen$, showing regions over which each conditional plan will be selected. b) $\alpha$-vectors of 3 conditional plans. c) Observation function and aggregate probabilities of each observation region for each state. d) Average discounted reward achieved over 10 trials of 100 simulated runs of 50 steps each, for different observation variances in the continuous 1D tiger problem.

## 5 Multi-dimensional observation space

In many applications, observations are composed of several sensor readings. For example, a mobile robot may have a number of sonar, laser or tactile sensors, which together give rise to multi-dimensional observations. When the observation space is multi-dimensional, analytically finding the regions and computing their aggregate probability will be difficult for many observation functions.

We examine two approaches. The first, discussed in Sect. 5.1, considers the special case of POMDPs with observations composed of conditionally independent variables. For this special case, it is possible to define an equivalent POMDP where the observation variables are processed sequentially in isolation, essentially reducing the observation space to one dimension. For arbitrary multi-dimensional observations, we present a general sampling-based technique in Sect. 5.2.

### 5.1 Conditionally Independent Observations

In some applications, the random variables corresponding to the individual sensor measurements may be conditionally independent given the last action executed and the state of the world. In this case, it is possible to factor the observation function into a product of small observation functions, one for each random variable (i.e., $\Pr(z_1, z_2|a, s') = \Pr(z_1|a, s') \Pr(z_2|a, s')$). For example, consider a mobile robot with sonars pointing forwards and to the side. The readings from each sonar are conditionally independent given the location of the robot and a map of the robot's environment.

This factorization can be exploited to process the observations sequentially. For $n$ conditionally independent observation variables, we divide each time step into $n$ sub-steps such that only one observation variable is observed per sub-step. This can be easily accomplished by constructing a POMDP with an additional state variable, *sub_step*, that keeps track of the current sub-step. The observation function encodes the probabilities of a single, but different, observation variable at each sub-step. The transition function is the same as in the original POMDP when *sub_step=1*, but becomes the identity function otherwise. The rewards are gathered and the discount factor applied only when *sub_step=1*.

When all observation variables are conditionally independent, this effectively reduces the dimensionality of continuous observations to one, allowing the approach of Sect. 4 to be used. For discrete observation variables, an exponential reduction is also achieved since the domain of a single variable is exponentially smaller than the cross-domain of several variables. Note however that the complexity of the equivalent POMDP remains the same since the reduction is achieved by multiplying the horizon and the number of states by $n$.

### 5.2 Sampling

For arbitrary multi-dimensional observations, an effective approximation technique for computing the aggregate probabilities consists of sampling. Recall from Sect. 3 that for each conditional plan $cp$, we can aggregate in one region $\mathcal{Z}_{cp}$ all the observations $z$ for which $cp$ yields the highest value (i.e.,$\beta_{cp}^{b,a}(z) \geq \beta_{cp'}^{b,a}(z)\ \forall cp' \in \mathcal{CP}$). Hence, for each $\langle a, s' \rangle$-pair, we sample $k$ observations from $pdf(z|a, s')$ and set $\Pr(\mathcal{Z}_{cp}|a, s')$ to the fraction of observations for which $\beta_{cp}^{b,a}(z)$ is maximal, breaking ties by favoring the conditional plan with the lowest index.

This sampling technique allows us to build an approximate discrete observation function $\Pr(\mathcal{Z}_{cp}|a, s')$ which can be used to perform a point-based backup for a set $\mathcal{CP}$ of conditional plans and a belief state $b$. The number of observations sampled is $k|\mathcal{S}||\mathcal{A}|$ for each point-based backup. The quality of the approximation improves with $k$. In particular, using Hoeffding's bound [9], we can guarantee that $\Pr(\mathcal{Z}_{cp}|a, s')$ has an error of at most $\epsilon$ with probability $1 - \delta$ when $k = ln(2|\mathcal{CP}|/\delta)/(2\epsilon^2)$. Interestingly, $k$ doesn't depend on the dimensionality (nor any other complexity measure) of the observation space. It depends only on the number of regions, which is at most the number of conditional plans. While this is true in a single DP backup, the number of conditional plans may increase exponentially (in the worst case) with the number of observations at each DP backup [12]. On the other hand, several algorithms [18; 8; 17; 24] can mitigate policy complexity by searching for good yet small policies represented by a bounded number of condi-

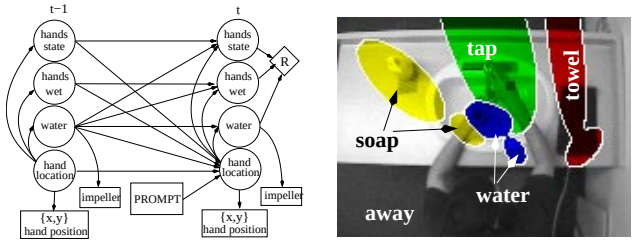
tional plans or $\alpha$-vectors. Perseus [24], a randomized point-based value iteration algorithm, is such an algorithm since the number of $\alpha$-vectors is bounded by the number of belief points initially sampled. Hence $k$ depends on policy complexity, which generally depends on observation complexity, but can be made independent by restricting policies to have a bounded representation.

This sampling technique can also be used for POMDPs with many discrete observations. In particular, when the observations are factored into several random variables, the number of observations is exponential in the number of variables, but as long as the number of conditional plans remains small, the number of samples will also be relatively small.

Note also that we can often weaken the dependency between the number of samples and the size of the action and state spaces. Instead of sampling $k$ observations from each of the $|\mathcal{A}||\mathcal{S}|$ densities $pdf(z|a,s')$, we can sample $j$ observations from one proposal distribution $p(z)$. This sample of $j$ observations can be used to approximate each $\Pr(\mathcal{Z}_{cp}|a,s')$ as follows. First, we assign a weight $pdf(z|a,s')/p(z)$ to each sampled observation $z$ to obtain an unbiased sample of $pdf(z|a,s')$. Then, for each conditional plan $cp$, we find the subset of sampled observations $z$ for which $\beta_{cp}^{b,a}(z)$ is maximal, and set $\Pr(\mathcal{Z}_{cp}|a,s')$ to the (normalized) sum of the weights of the observations in that subset. The number of sampled observations $j$ necessary to guarantee an error of at most $\epsilon$ with probability $1-\delta$ depends on how similar the proposal distribution $p(z)$ is with each density $pdf(z|a,s')$. When the proposal is relatively similar to each of the densities then $j$ tends to be close to $k$, significantly reducing the dependencies on $|\mathcal{A}|$ and $|\mathcal{S}|$. However, as the differences between the proposal and each of the densities increase, $j$ also increases and may become arbitrarily large. In Sect. 6, we use $pdf(z|b,a)$ as a proposal distribution.

## 6 Experiments

This section presents experiments with a POMDP that assists people with cognitive difficulties to complete activities of daily living. Focusing on the task of handwashing, we present a simplified POMDP for guiding patients with verbal prompts as they wash their hands. The goal of such a system is to minimize the human caregiver burden, and is part of an ongoing research initiative applying intelligent reasoning to assistive living [4]. In this paper, we present results from simulations of our methods on a simplified POMDP model for the handwashing task. Fig. 3(a) shows the graphical model of the handwashing POMDP. The POMDP's actions are the verbal prompts (corresponding to the canonical steps of handwashing: *turn on water*, *wet hands*, *use soap*, *dry hands* and *turn off water*) and a *null* action where the system waits. The states are defined by the variables *hands_state*, which can be {*dirty, soapy, clean*}, *hand_location*, which can be {*away, tap, water, soap, towel*}, *hands_wet*, which can be {*wet, dry*}, and *water*, which can be {*on, off*}. We assume the hands start *dirty* and *dry*, and the goal is to get them *clean* and *dry*, which can only happen if they become *soapy* and *wet* at some intermediate time. The water starts *off* and must be *off* for task completion. The cost of a prompt is 0.2, and a large reward




(a) (b)

Figure 3: a) POMDP model of handwashing. b) Example image showing the regions induced by the observation function alone, $\max_i P(x,y|hand\_location=i)$

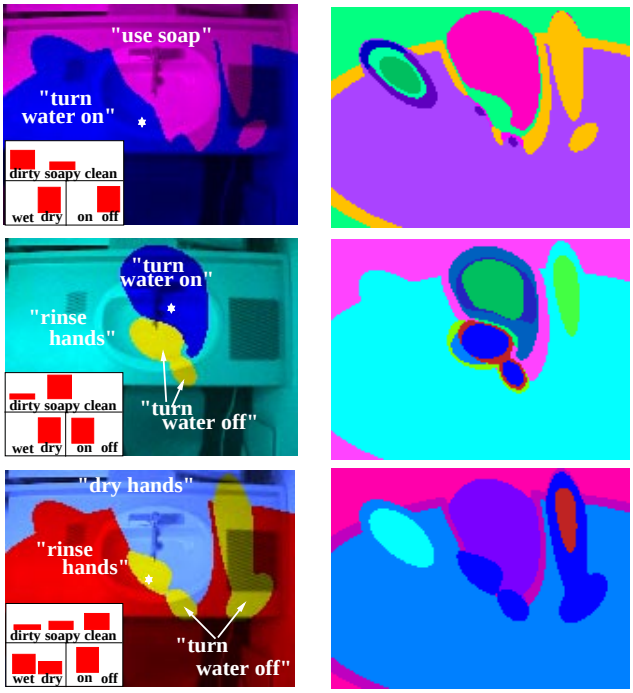
(20) is given when the hands are dry and clean and the water is off. Smaller rewards are given for progressions through the task: if the hands are clean (1), soapy (0.5) and wet (0.2), but other variables are not as in the final goal state. The discount was $\gamma = 0.85$.

We model the system as being equipped with an impeller in the water pipe, which returns 0 when there is no water flowing and 1 when the water is on full. The sensor's noise is zero-mean Gaussian with standard deviation $\sigma = 0.4825$, which gives it an 85% accuracy (when the most likely state of *water* is considered). The position of the hands in the horizontal plane is measured using a camera mounted in the ceiling above the sink, connected to a computer vision system that returns an estimate of the $\{x,y\}$ position of the patient's dominant hand in the image using skin color [14]. Fig. 3(b) shows an example image from the camera. The observation function that relates these measured hand positions to the actual *hand_location* was learned from a set of data taken from the computer vision system. An actor simulated the repeated handwashing trials for about 10 minutes. The vision system tracked and reported the $\{x,y\}$ position of the right (dominant) hand, while a researcher annotated the data with the actual *hand_location*.[5] The functions $P(x,y|hand\_location=i)$ were then learned by fitting a mixture of Gaussians to the data annotated with the value $hand\_location = i$.[6] The mixture models were fit using a K-means initialization followed by the expectation-maximization algorithm. Figure 3(b) shows the most likely *hand_location*s for each $\{x,y\}$ position induced by the learned mixtures of Gaussians. The water flow observation function was not learned from data.

The water flow and hand position readings yield a 3D observation space. Although water flow and hand positions are conditionally independent, the $\{x,y\}$ coordinates of the measured hand positions are dependent. Hence, we cannot process the observations sequentially as suggested in Sect. 5.1 and must resort to the sampling technique of Sect. 5.2. We extended Perseus [24] with the sampling technique described

[5] The vision system only reports the hand position every 2 seconds, or when the hand location changes and is located consistently for 5 frames (1 second).

[6] The model orders were selected by minimizing the minimum description length (MDL): $-log(P(z|hand\_location)) + \frac{1}{2}MlogN$, where $M$ is the number of parameters and $N$ is the number of data points. This yielded between 2 mixture components for each state. The *away* state was fixed to have 1 mixture component.




(a) (b)

Figure 4: Example regions during a simulated trial at three stages (from top to bottom, after 1, 3 and 4 actions): (a) shows the best actions to take, the current belief state, and the current location of the hand (star); (b) shows the regions $\mathcal{Z}_{cp}$, each different shade corresponding to a different conditional plan that will be followed if an observation is made in the region.

in Sect. 5.2 to compute an approximate discrete observation function at each point-based backup. More precisely, 100 observations were sampled before each point-based backup to approximate the aggregate probabilities of each observation region induced by some conditional plan. We ran the algorithm for 50 iterations with 227 sampled belief states. For comparison purposes, we also ran the original Perseus algorithm with a fixed discrete observation function obtained by partitioning the plane of hand positions according to the regions shown in Fig. 3(b) and by partitioning water flow readings in two regions at $0.5$. This is a natural discretization that arises from the observation function by considering the regions where each $pdf(x, y|hand\_location)$ and $pdf(impeller|water\_flow)$ are highest. The computed policies were then simulated, and actions were selected by monitoring the belief state and computing the best conditional plan $\arg\max_{cp} \sum_s b(s)\alpha_{cp^*}(s)$. The discounted rewards (averaged over 10 trials of 100 simulated runs of 50 steps each) of the policies obtained by dynamic partitioning are $13.7 \pm 1.4$, while those for fixed partitioning are $4.7 \pm 0.3$, showing that our sample-based dynamic partitioning technique outperforms the fixed discretization technique. The final conditional plans were represented with $64$ $\alpha$ vectors for our dynamic algorithm, and 77 $\alpha$ vectors for the fixed discretization.

Figure 4 shows examples of the dynamic partitions found by our technique at three stages during a simulated trial. Since we cannot show 3D partitions, we show the 2D partitions of the $\{x, y\}$ plane (ignoring the water flow sensor). At the beginning of the trial (top row, stage 1), the system will either prompt to use the soap or turn the water on, based on where it sees the hands. At this stage, the regions $\mathcal{Z}_{cp}$ mainly distinguish the areas surrounding the soap and the taps, since these are the usual first steps in handwashing. Once the hands are believed to be soapy and the water on (stage 3, middle row),the system will prompt to rinse the hands, unless the patient has rinsed their hands or has turned the water off, in which case the prompt will be to turn the water off or on, respectively. We see in Figure 4(b) (middle row) that there are many regions now in the areas near the tap or under the water. This is the most uncertain area for this system (see Figure 3(b)), calling for many different conditional plans. Finally, at stage 4 (bottom row), the system believes the hands are clean, and will prompt the user to dry their hands or turn the water off. In this case, fewer possibilities remain, and so there are fewer regions in Figure 4(b) (bottom row).

## 7 Conclusion

Exploiting the fact that observations are useful only to the extent where they lead to different courses of actions, the paper describes how to dynamically partition observation spaces without any loss of information based on the current policy. For policies with a small number of conditional plans, observations can be aggregated in a small number of regions corresponding to the relevant observation features of the application domain. The region-based observation function can generally be constructed by numerical root-finding and integration algorithms for uni-dimensional observations or multi-dimensional observations composed of conditionally independent variables. For general multi-dimensional observations a general sampling technique was also described and demonstrated on a realistic assisted living task.

Note that the dynamic partitioning technique proposed in this paper is tightly integrated with point-based backups. More precisely, a lossless dynamic partition of the observation space can be computed only with respect to a given belief state and a set of $\alpha$-vectors. As a result, our technique cannot be integrated with algorithms that do not use point-based backups (e.g., Incremental Pruning [26], Bounded Policy Iteration [18]). Furthermore, it cannot be integrated with the linear programs that find belief points prior to point-based backups in the Witness algorithm [12]. At the moment, full integration is only possible with Linear Support [6], PBVI [17], and Perseus [24] since these algorithms make use of the observation function only in point-based backups. Dynamic lossless observation partitioning for a broader range of algorithms is a possible direction for future research.

This paper tackles POMDPs with continuous observations, but discrete states. As mentioned earlier, such POMDPs are common in user modeling, event recognition and spoken-dialog systems, since the observations correspond to continuous sensor readings and the states are abstract discrete features. Note also that our dynamic partitioning technique doesn't require the state space to be discrete. In fact, Porta et. al [11] recently proposed an extension to Perseus that can handle continuous state spaces. Point-based backups are performed in a similar fashion, but given the continuous nature

of the state space, $\alpha$-functions are generated instead of $\alpha$-vectors. Integrating our dynamic partitioning technique with such continuous point-based backups should be possible and is subject to future research.

Our current work in using POMDPs for assistive living tasks involves learning model structure and user behaviors from sequence data, rather than imposing our own structure on tasks. The POMDP models we learn have observation functions which are themselves dynamic Bayesian networks (DBNs) with video frame observations at each time step, leading to a hierarchical model. Preliminary work along these lines is reported in [10]. We wish to use the techniques we have described in this paper both to solve these POMDPs and to learn models of human behaviors from video sequence data. Another potential research direction includes the exploration of automated feature detection in application domains such as image processing and speech recognition by policy-directed observation aggregation.

**Acknowledgements**

The authors wish to thank Alex Mihailidis for help with the handwashing data, Darius Braziunas for pointing out some inconsistencies in some early experiments, Nikos Vlassis for helpful comments and Jason Williams for early discussions. The first author gratefully acknowledges the support of Intel Corporation and the American Alzheimer Association.

## References

[1] K. J. Åström. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

[2] D. Aberdeen and J. Baxter. Scaling internal-state policy-gradient methods for POMDPs. In *ICML*, pages 3–10, Sydney, Australia, 2002.

[3] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, Belmont, MA, 1996.

[4] J. Boger, P. Poupart, J. Hoey, C. Boutilier, G. Fernie, and Alex Mihailidis. A decision-theoretic approach to task assistance for persons with dementia. In *Proc. IJCAI*, Edinburgh, 2005.

[5] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *AAAI*, Seattle, WA, 1994.

[6] H.-T. Cheng. *Algorithms for Partially Observable Markov Decision Processes*. PhD thesis, University of British Columbia, Vancouver, 1988.

[7] T. Darrell and A. P. Pentland. Active gesture recognition using partially observable Markov decision processes. In *IEEE Intl. Conf. on Pattern Recognition*, Vienna, Austria, 1996.

[8] Z. Feng and E. Hansen. Approximate planning for factored POMDPs. In *Proc. ECP*, Toledo, Spain, 2001.

[9] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[10] Jesse Hoey, Pascal Poupart, Craig Boutilier, and Alex Mihailidis. Semi-supervised learning of patient-caregiver interactions using partially observable Markov decision processes. Working paper, 2005.

[11] M. Spaan J. Porta and N. Vlassis. Value iteration for continuous-state POMDPs. Technical Report IAS-UVA-04-04, Informatics Institute, University of Amsterdam, 2004.

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

[13] N. Meuleau, L. Peshkin, K.-E. Kim, and L. P. Kaelbling. Learning finite-state controllers for partially observable environments. In *UAI*, pages 427–436, Stockholm, 1999.

[14] A. Mihailidis, B. Carmichael, and J. Boger. The use of computer vision in an intelligent environment to support aging-in-place, safety, and independence in the home. *IEEE Trans. on Information Technology in Biomedicine (Spec. Issue on Pervasive Healthcare)*, 8(3):1–11, 2004.

[15] M. Montemerlo, J. Pineau, N. Roy, S. Thrun, and V. Verma. Experiences with a mobile robotic guide for the elderly. In *AAAI*, pages 587–592, Edmonton, AB, 2002.

[16] A. Y. Ng and M. Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *UAI*, pages 406–415, Stanford, CA, 2000.

[17] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: an anytime algorithm for POMDPs. In *IJCAI*, Acapulco, Mexico, 2003.

[18] P. Poupart and C. Boutilier. Bounded finite state controllers. In *NIPS*, Vancouver, BC, 2003.

[19] P. Poupart and C. Boutilier. VDCBPI: an approximate scalable algorithm for large POMDPs. In *NIPS*, Vancouver, BC, 2004.

[20] N. Roy and G. Gordon. Exponential family PCA for belief compression in POMDPs. In *NIPS*, pages 1635–1642, Vancouver, BC, 2002.

[21] N. Roy, J. Pineau, and S. Thrun. Spoken dialog management using probabilistic reasoning. In *ACL*, Hong Kong, 2000.

[22] R. Smallwood and E. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[23] S. Thrun. Monte Carlo POMDPs. In *NIPS*, pages 1064–1070, Denver, 1999.

[24] N. Vlassis and M. T. J. Spaan. A fast point-based algorithm for POMDPs. In *Proc. Belgian-Dutch Conference on Machine Learning*, Brussels, Belgium, 2004.

[25] J. Williams, P. Poupart, and S. Young. Using factored Markov decision processes with continuous observations for dialogue management. Technical Report CUED/F-INFEG/TR.520, Cambridge University, Engineering Department, 2005.

[26] N. Zhang and W. Liu. Planning in stochastic domains: Problem characteristics and approximation. Technical Report HKUST-CS96-31, Hong Kong University of Science and Technology, 1996.

[27] N. Zhang and W. Zhang. Speeding up the convergence of value-iteration in partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 14:29–51, 2001.