The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Minimum Uncertainty Robot Path Planning using a POMDP Approach

Salvatore Candido and Seth Hutchinson

Abstract— **We propose a new minimum uncertainty planning technique for mobile robots localizing with beacons. We model the system as a partially-observable Markov decision process and use a sampling-based method in the belief space (the space of posterior probability density functions over the state space) to find a belief-feedback policy. This approach allows us to analyze the evolution of the belief more accurately, which can result in improved policies when common approximations do not model the true behavior of the system. We demonstrate that our method performs comparatively, and in certain cases better, than current methods in the literature.**

I. INTRODUCTION

We present a new method for finding minimum uncertainty paths for mobile robots. This problem has previously been considered in [1]–[3] for the specific case of finding minimum uncertainty paths for mobile robots localizing with beacons. We consider the same scenario, but compute policies using a new approach that models the system as a partially-observable Markov decision process (POMDP) whose cost criterion minimizes uncertainty in the robot's position. Since computing optimal policies for this problem is intractable [4], we use a sampling-based method to construct a belief-feedback policy.

Most methods for dealing with uncertainty while planning robot motions construct plans in a configuration space that has been augmented to represent uncertainty. Such approaches necessarily restrict the type and degree of uncertainty that can be considered, since uncertainty is represented by an a posteriori probability function on the state space (beliefs). Typically, the posterior is assumed to be Gaussian, parameterized by mean and covariance. In contrast, our approach operates directly in the belief space (i.e., the space of all possible posteriors). We represent beliefs using particle filtering methods, thus allowing the consideration of arbitrary posteriors, given an appropriate number of particles. This capability is essential, e.g., when robots operate in environments that contain obstacles, or when uncertainty is sufficiently large to cause difficulty with data association.

Furthermore, typically, methods that plan in the (possibly augmented) configuration space do not consider variation in possible future observations. This is because considering multiple future observations at each stage causes an exponential growth in the number of sample paths that must be considered. To cope with this, it is typical to consider only the maximum-likelihood (ML) observations at future stages. We utilize hyper-particle filtering [5] which allows us to account for uncertainty under many possible sequences of observations the robot may encounter, rather than committing to the ML contingency. We heuristically explore the belief space using local policies, and optimize over a graph representing these trajectories through the belief space. Since we are optimizing in the belief space, if a sufficient number of observations are sampled, those trajectories will be representative of the evolution of the POMDP, not just one contingency.

We consider a standard model for a mobile robot with localization beacons (Section III). The robot has configuration in $SE(2)$ and a probabilistic motion model. Observations, correlated with state, arise from a set of noisy beacons, where the accuracy and precision of measurements from a beacon improve as the robot moves closer to the position of the beacon. Given start and goal configuration, the objective is to minimize an uncertainty measure along the path to the goal.

This system can be modeled as a POMDP with continuous state, control, and observation spaces. We demonstrate that by using our POMDP planning method, originally proposed by the authors in [6], we can find policies comparable to and, in some cases, better than other existing methods. This is the case even when restrictive assumptions are imposed (e.g., Gaussian noise, no obstacle constraints). In short, our method is better able to evaluate uncertainty in portions of the space where the transition functions exhibit nonlinear behavior and the approximation of the belief as a Gaussian distribution is not faithful to the true pdf. Furthermore, our proposed method can be generalized to find minimum uncertainty plans for robot systems with more complicated configuration spaces or with non-Gaussian process and observation models.

A. Related Planning with Uncertainty Research

Consideration of uncertainty created by an uncertain process model was first combined with sampling-methods in [7] to predict the behavior of a system. Rather than just predicting, in [8]–[10], attempts were made to extend sampling-based algorithms to plan for systems with uncertainty. The work in [8] considers uncertainty in the robot's motion model, while [9], [10] consider uncertainty in the map of the environment. The Sampling Hyperbelief Optimization Technique [11] also constructs a graph based representation of trajectories for uncertain systems. However, it differs from these other methods by operating in the space of probability function over the belief space.

This material is based in part upon work supported by the National Science Foundation under award CNS 0931871.

S. Candido is in the Department of Electrical and Computer Engineering at the University of Illinois. candido@illinois.edu

S. Hutchinson is a professor of Electrical and Computer Engineering at the University of Illinois. seth@illinois.edu

Other work in the robotics community generates plans to minimize uncertainty for specific robot tasks. The active localization algorithms of [12] and [13] make robot localization more effective by specifically considering expected uncertainty of the localization algorithm while planning the next control the robot will receive. These algorithms generate a control to minimize uncertainty at the next stage, and do not optimize over a path or specify a goal position.

The Belief Roadmap planner (BRM) [1], [2], [14] and the method of [3] have similar goals and environment models to our work. Like [1], [14], we use a sampling-based planner, e.g., [15], in the robot's workspace to heuristically construct a set of local policies. However, [1]–[3], [14] restrict the set of possible policies to make the optimization procedure feasible and commit to a particular representation of belief (Gaussian). These methods use Extended and Unscented Kalman Filters to approximate the representation of belief as a multi-variate normal distribution over the state space. We differ by using a sequential Monte Carlo representation that can more closely model the true pdf, if a sufficient number of particles are used. Furthermore, the above methods control the growth of possible future belief states by considering only the ML observation when evaluating policies. The main advantage of our method is that we explicitly attempt to characterize many possible future observations at every stage, and can in some cases better gauge the costs different policies will incur.

There is a large volume of work related to solving both discrete and continuous POMDPs. We refer the reader to [6] for citations that are directly related to the POMDP optimization algorithm we utilize. Particularly relevant to our method is [16], which discusses POMDP optimization in continuous spaces.

II. POMDP MODEL AND OPTIMIZATION

In this section, we briefly review POMDPs, primarily to establish notation and provide the background necessary to motivate the method to approximate the value function. For a more thorough discussion of the POMDP model see [17] or [18].

A. POMDP Framework

A POMDP is comprised of a state space $\mathcal{X}$, set of controls $\mathcal{U}$, a process model, set of observations $\mathcal{Y}$, and sensor or observation model. The belief b_t is the posterior probability function over $\mathcal{X}$ conditioned on the information state at t, a set containing the initial pdf and all controls and observations prior to stage t. We rely on the belief to act as a sufficient statistic of the information state [19]. In practice, we usually cannot represent the exact pdf, so b_t will refer to an approximation of the exact belief in the belief space $\mathcal{P}_b$, the space of all pdfs over $\mathcal{X}$. Using this quantity as the state vector of the POMDP, we can treat the original system as an MDP evolving in the belief space.

This process has both a *transition probability function* that corresponds to Bayesian prediction, and a *transition observation function* that corresponds to Bayesian update. Putting these together, we arrive at the *belief transition operator* for the POMDP

$$b_{t+1} = \phi_{y,u} b_t. \tag{1}$$

For a more detailed discussion of MDP's with continuous state variables, refer to [20]. The belief transition operator $\phi_{y,u}$ functions as the process model for the belief space MDP and observations are the random variable in the process evolution.

We consider cost functions that are separable into a one-stage cost function $c_b : \mathcal{U} \times \mathcal{P}_b \to \mathbb{R}$ and a terminal cost function $c_{b\bar{t}} : \mathcal{P}_b \to \mathbb{R}$. Although different formulations of cost are available, we consider an infinite-horizon total cost criterion with retirement option. Thus, the cost under policy π is

$$J(b_0, \pi) = \mathrm{E}\left[\sum_{t=1}^{\bar{t}} c_b(\pi(b_{k-1}), b_t) + c_{b\bar{t}}(b_{\bar{t}}) \middle| b_0\right]$$

where the expectation is taken over the joint pdf over the sequence of observations from stages 1 to $\bar{t}$. At every stage, the policy may choose to "retire" with cost specified by $c_{b\bar{t}}(\cdot)$, where $\bar{t}$ denotes the a priori unknown retirement stage.

B. Switched Policy Optimization

Our minimum-uncertainty robot planning method is an anytime policy-improvement algorithm based on an approximation of the stochastic Hamilton-Jacobi-Bellman (sHJB) equation. Our approach will involve utilizing a set of local policies that specify multi-stage trajectories through the hyperbelief space. We choose the policy to be a switched policy whose modes and switching conditions are defined by a set of local policies. We model the robot system as a partially-observable Markov decision process (POMDP) and use a diffusion-based approach to explore the search tree with each expansion following a trajectory of a particular local policy. The algorithm builds a directed graph in the belief space, representing the evolution of the POMDP. This data structure is used to compute both a switched policy and the expected cost to the system under this policy. However, since we are using closed-loop feedback control policies at every stage, this is not a straightforward conversion of the original POMDP to a temporally abstracted POMDP using macro-actions. To avoid the computational intractability of optimizing the policy control by control [4], we optimize the policy in a coarse fashion.

We introduced this algorithm in [6], and demonstrated its effectiveness in planning the policy of a team of mobile robots attempting to perform a coordinated task, manipulating (extinguishing) a stochastic process that loosely models the spread of a fire. Please refer to [6] for more details on this algorithm and research publications related to the core optimization algorithm.

III. PROBLEM MODEL

In this section, we discuss the robot model and how it is simulated during policy optimization. Because the system has continuous $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{U}$, we cannot finitely parameterize

$\mathcal{P}_b$ and, in general, there is no exact representation of belief that does not grow in complexity with the number of stages executed. We represent a belief by a set of particles, an established procedure for approximating a function [17], [21]. With this approximation, we work with a pmf over a finite set of support points from the belief pdf, but do not discretize the state, observation, or control spaces, i.e., the support points are neither fixed nor representative of a region of the state space.

A. Robot and Environment Model

We use the model used in [3], [14], a mobile robot with configuration in $SE(2)$, moving through an environment with multiple beacons that provide increasingly reliable measurements with decreasing distance to the beacon. We denote the robot's state as $x \in SE(2)$, with $x = [x^1, x^2, x^3]$ where $[x^1, x^2] \in \mathbb{R}^2$ is the robot's position and, $x^3 = \theta \in \mathcal{S}^1$ is the robot's orientation. The robot's transition model is a discrete version of the standard unicycle model, with additive linear Gaussian noise on the system input, e.g., the desired translational and rotational movement. The noise is a random vector drawn from a stationary, mean zero normal distribution with a diagonal covariance matrix.

Noisy measurements of the position of the robot come from a collection of distance beacons. At every stage, the robot receives a measurement from every beacon in the workspace, but the quality of the information conveyed varies with the robot's distance from the beacon. Specifically, the i^{th} beacon reports a measurement y^i that is a random variable with distance-dependent bias in mean and distance-dependent variance. We model the dependence as a linear function, specifically

$$\mu_b^i(d) = \mu_b^m d + \mu_b^b, \qquad \sigma_b^i(d) = \sigma_b^m d + \sigma_b^b$$

where $d = ||p - p_{\text{b}}^i||_2$ and $p_{\text{b}}^i \in \mathbb{R}^2$ is the location of the i^{th} beacon. Thus, the measurement from the i^{th} beacon is modeled as a random variable drawn from $\mathcal{N}(\mu_b^i(d), \sigma_b^i(d))$, a normal distribution with mean $\mu_b^i(d)$ and variance $\sigma_b^i(d)$. An observation y_t consists of a random vector of measurements, where the i^{th} entry is y^i.

This is not a linear Gaussian system, and the belief over the state space with respect to the information history will typically not be a normal pdf. In fact, using the approximation of a normal pdf is often poor because although all disturbances are drawn from normal distributions, the process model is nonlinear. This will cause the distance between the belief and any normal distribution to increase as t increases. Although the pdf of measurements is normal along the line extending from the beacon to the robot, when extending the pdf $\mathrm{P}\left[x_t | y_t\right]$ over the workspace, level sets of constant likelihood form circles centered on the beacon. Thus, the observations provide a circular distortion on the belief, particularly when close to beacons. Other nonlinearities, e.g., obstacles in the workspace, compound this problem.

B. Belief Approximation and Transition Functions

We use a sequential Monte Carlo method [17], [21] and approximate b_t as a collection of weighted particles. The particles are propagated through the POMDP's transition function using a particle filter to maintain an approximation of the pdf over $\mathcal{X}$ in a manner consistent with the POMDP. The particle-filtered representation is also a belief, but is not equal to exact belief conditioned on the information state.

Let $b_t = \{(x^{(i)}, w^{(i)})\}_{i=1,\cdots,m}$ be a collection of states and associated weights, i.e., particles. We use the Monte Carlo localization algorithm of [22], which was designed for mobile robot localization, to approximate the belief. We can approximate the probability transition function for a particle $x^{(i)}$ by sampling the process model according to the distribution of process noise. The probability observation function re-weights every particle in the set according to an observation y_t. To re-weight the particle with $x^{(i)}$, we must find the likelihood of y_t conditioned on the state of the system being $x^{(i)}$. For every $x^{(i)}$ in b_t,

1) Compute the distance to every beacon. For the j^{th} beacon, this is $d(i,j) = ||x^{(i)} - x_{\text{b}}^j||_2$.
2) Compute the expected value (mean) of the random vector y_t, assuming $x^{(i)}$ refers to the true position of the robot. Thus, for every beacon $\mu_b^j = \mu_b^m d(i,j) + \mu_b^b$ and $\mu_y^{(i)} = \mathrm{E}\left[y_t | x^{(i)}\right] = [\mu_b^1, \mu_b^2, \cdots, \mu_b^N]'$.
3) Re-weight each particle by the likelihood of y_t under $\mathrm{P}\left[y_t | x^{(i)}\right]$. Thus, $w^{(i)}$ is the likelihood of y_t under a normal distribution with mean $\mu_y^{(i)}$ and diagonal covariance matrix with the quantity $\sigma_b^m d(i,j) + \sigma_b^b$ on the j^{th} entry of the diagonal.

Finally, b_t is normalized so that $\sum_i w^{(i)} = 1$.

C. Local Policies

The local policies $\overline{\pi}^{x_g} \in \overline{\Pi}$ with goal configuration $x_g \in \mathcal{X}$ are a parameterized class of self-stopping, belief-feedback policies, where x_g is the target position the robot attempts to attain. Each local policy consists of a belief-feedback policy $\pi^{x_g} : \mathcal{P}_b \to \mathcal{U}$ and a stopping function $a^{x_g} : \mathcal{P}_b \times \mathbb{Z}^+ \to \{0, 1\}$. Given a goal position x_g,

$$u_T = \operatorname{atan2}(x_g^2 - \mathrm{E}_b\left[x^2\right], x_g^1 - \mathrm{E}_b\left[x^1\right]) - \mathrm{E}_b\left[x^3\right]$$
$$u_D = \min(||[\mathrm{E}_b\left[(x^1, x^2)\right] - (x_g^1, x_g^2)||_2, u_{D,max})$$

where $u = [u_D, u_C]'$ refers to the control action that is specified by the policy. Please note that the superscripts refer to components of the vector, and not exponents or moments. Essentially, this class of policies drives the expected value of the robot's position towards a desired goal position in the workspace. The policy stops when the mean of the belief gets sufficiently close to the target point, less than ϵ_ϕ or after the policy has executed a set number of stages $t_{\max}$.

D. Observation Sampling Function

Since observations are not available at planning time, we optimize the policy based on sampling $\mathcal{Y}$ based on the likelihood conditioned on belief, $\mathrm{P}\left[y_t | b_t\right]$. Consider the i^{th} component of the random vector y_t. Using the law of total probability, we can compute the exact (continuous) belief as

Algorithm 1: Expand Method

Input: b - belief state to expand,
$\overline{\pi}$ - local policy to use at b
Output: c - expected cost to go using $\overline{\pi}$,
β - destination hyperbelief

$c \leftarrow 0$, $t \leftarrow 0$, $\alpha \leftarrow 1$, $\beta_f \leftarrow \emptyset$, $\beta_0 = \{(b, 1)\}$
while $\alpha > 0$ **do**
  $t \leftarrow t + 1$
  $(c_t, \beta_t) \leftarrow$ HyperParticleFilter$(\beta_{t-1}, \overline{\pi})$
  $c \leftarrow c + \alpha c_t$, $\alpha' \leftarrow \alpha$
  foreach $(b^{(i)}, w^{(i)}) \in \beta_t$ **do**
    if $\overline{\pi}.a(b^{(i)}, t) = 1$ **then**
      $\alpha' \leftarrow \alpha' - \alpha w^{(i)}$
      $\beta_t \leftarrow \beta_t \ \{(b^{(i)}, w^{(i)})\}$
      $\beta_f \leftarrow \beta_f \cup \{(b^{(i)}, \alpha w^{(i)})\}$
  Normalize weights of β_t to 1
  $\alpha \leftarrow \alpha'$
Return (c, β_f)

shown in (2). With a particle-filtered representation of b_t, this pdf is best approximated by (3).

$$\mathrm{P}_{\Gamma_{m_t}}\left[y_t^i = y | b_t\right] = \int_{x \in \mathcal{X}} \mathrm{P}_{\Gamma_{m_t}}\left[y_t^i = y | x\right] \mathrm{P}_{b_t}\left[x\right] dx \quad (2)$$

$$\approx \sum_j \mathrm{P}_{\Gamma_{m_t}}\left[y_t^i = y | x^{(j)}\right] w^{(j)} \quad (3)$$

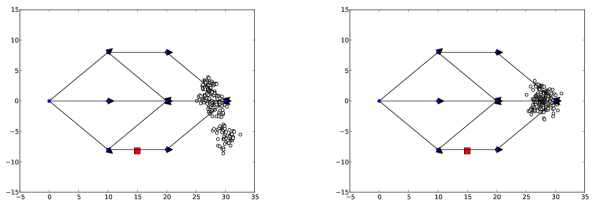
where Γ_{m_t} is the distribution governing the noise random vector in the observation model. Sampling from this distribution can be accomplished by sampling $x^{(j)}$ with probability $w^{(j)}$. Then, sample y_t^i from $\mathrm{P}_{\Gamma_{m_t}}\left[y_t^i | x^{(j)}\right]$.

IV. MINIMUM UNCERTAINTY PLANNING

In our planning method, one of the main points of emphasis was the integration of domain knowledge into the planner. The assumption of problem-specific domain knowledge is particularly applicable in robotics, i.e., often the local structure of nearly optimal trajectories is known. The combination of local policies with hyper-particle filtering allows us to develop temporal abstraction, which can be used search the reachable belief space in depth and produce effective policies without requiring an impractical amount of computation.

The core idea of this method is to use paths prescribed in the workspace for deterministic robots as a heuristic, but to evaluate the effects and to plan in the belief space using value function approximation. We begin with a graph in the workspace, where each vertex corresponds to a point in $\mathbb{R}^2$. One of these vertices should be located at the mean of the position for the initial belief b_0.

The policy and belief sampling function is implemented as a queue data structure. When a new belief b^i is added to $\mathcal{B}$, it is mapped to a vertex in the workspace graph by choosing the vertex whose position in the workspace is closest to the mean of the position of the belief. Then, for every outgoing edge from that vertex, we add the tuple $(b^i, \overline{\pi}^{x_g})$ where x_g is the location of the target vertex. Initially, the queue is populated by adding b_0 to $\mathcal{B}$. When the queue is empty, the algorithm terminates.



(a) Possible terminal belief using M_2 (b) Possible terminal belief using M_3

Fig. 1. Notable belief states

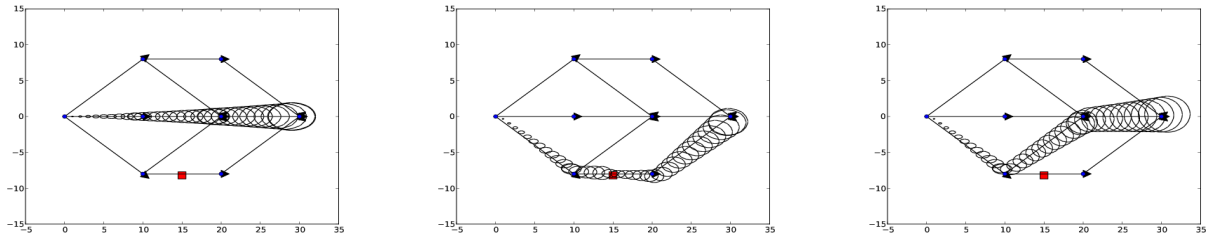
This policy choice assumes that, often, the majority of the probability density, when projected onto the workspace, congregates around vertices. In the context of the model we have explored, this assumption remained valid, but it is possible for a system with more stochasticity that this heuristic may produce undesirable results. At that point, heuristics based on data structures built for the configuration space will most likely not produce effective belief-feedback local policies.

New beliefs are added to $\mathcal{B}$ as a result of the Expand operation. This operation evaluates the effect of using $\overline{\pi}^{x_g}$ at b^i. We use an augmented version of hyper-particle filtering [5], which is shown in Algorithm 1. The belief b^i is propagated through $\phi_{y,u}$, i.e., (1), where u is chosen by $\overline{\pi}^{x_g}$ and y is sampled using the observation sampling function (Section III-D). This occurs for multiple particles, in parallel, producing an estimate of the expected cost and hyperbelief resulting from using the $\overline{\pi}^{x_g}$ at b^i. Note that particles in the hyper-particle filter may trigger the stopping condition at different stages, so additional bookkeeping is required.

V. EXPERIMENTAL RESULTS

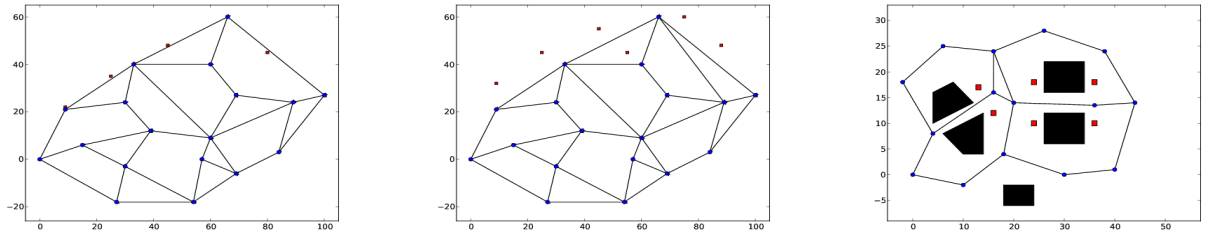
We tested our approach using a mobile robot simulation. After a short discussion of our experimental procedure, we present a very small example that demonstrates how a particle-filtered representation of belief and a POMDP approach is able to improve results over an approximated-Gaussian search of the roadmap. Next, we present larger examples that one might find in a realistic mobile robot scenario.

For purposes of comparison, we evaluated three methods. Method M_1 selects the distance optimal path through the graph, and is provided for a baseline comparison with a plan that does not explicitly consider uncertainty. Method M_2 approximates the belief state as a Gaussian random vector using the Extended Kalman Filter and samples the ML observation at every stage. This produces the same result as the method of [14]. Finally, our proposed method is referred to as M_3. From each method, we generate a belief-feedback policy. Using s sample path simulations, i.e., starting from b_0 we apply u_k generated by the appropriate policy and sample y_k with probability specified by $\mathrm{P}_{\Gamma_m}\left[y_k | b_k\right]$, we



(a) Path chosen by M_1 (b) Path chosen by M_2 (c) Nominal path chosen by M_3

Fig. 2. Approximated Gaussian, ML observation estimates of covariance.



(a) Environment 2 (b) Environment 3 (c) Environment 4

Fig. 3. Workspaces, beacons, and roadmaps for experiments.

then are able to evaluate the effectiveness of the policies by comparing the mean and variance of the costs of the sample paths.

We compared the three methods using a cost function that measures uncertainty at the goal,

$$J^{\pi}(b_0) = \mathrm{E}\left[\ ||\Sigma_N||_F^2 \ \middle| b_0\right]$$

where Σ_k is the sample covariance matrix of $[x^1, x^2]'$ conditioned on b_k, $||\cdot||_F^2$ denotes the square of the Frobenius norm, which in this case is equal to the sum of squared eigenvalues of the matrix, and N is the retirement stage. Many other cost metrics are available and can be used interchangeably this metric in our method.

Consider the example roadmap pictured in Fig. 2. The points represent roadmap vertices and the lines represent edges. The starting belief has the robot beginning at the leftmost vertex, with probability 1. The desired goal is the rightmost vertex. The beacon is represented by the red square. Fig. 2(a)-(c) show the covariance estimates along the graph paths chosen by M_1, M_2, and M_3, respectively. When close to a beacon, a measurement may have the effect of splitting a uni-modal probability distribution into a multi-modal one. This is demonstrated in Fig. 1(a), picturing a final belief from a sample path simulation using the M_2 optimal path. However, taking a path near to the beacon, but sufficiently far to minimize the nonlinear effects of the observation avoids this difficulty, e.g., Fig. 1(b). In fact, M_3 recognizes this during the planning process and decides to (nominally) use the path pictured in Fig. 2(b). This improves performance of $\mu_3, \sigma_3^2 = (3.471, 4.054)$ versus $\mu_1, \sigma_1^2 = (4.046, 6.605)$ and $\mu_2, \sigma_2^2 = (4.428, 18.065)$, where μ_i, σ_i^2 are the mean and variance of the costs of the sample path simulations using the policy specified by M_i.

A larger example is shown in Fig. 3(a)-(b). These examples highlight the same problem with nonlinearities close to beacons as the small example. For the environment of Fig. 3 (a), we have, $\mu_3, \sigma_3^2 = (4.921, 14.092)$ versus $\mu_1, \sigma_1^2 = (5.186, 18.029)$ and $\mu_2, \sigma_2^2 = (6.166, 42.275)$. Note that not only is the mean lower for M_3, using M_2 will also result in a higher variance, i.e., the cost along that path with vary more with respect to different sequences of observations.

For the environment of Fig. 3 (b), the M_2 nominal path through the graph (the path closest to the beacons) is, in fact, optimal. This is reflected by the similarity between μ_2 and μ_3, with $\mu_2, \sigma_2^2 = (2.749, 4.278)$ versus $\mu_3, \sigma_3^2 = (2.695, 3.306)$ as compared to $\mu_1, \sigma_1^2 = (3.473, 6.568)$. The mean for M_3 is slightly lower, due to the fact that the policy produced is a true belief-feedback policy. For different belief stages, e.g., belief states with means centered on different sides of a vertex, the policy may choose to take different graph paths. This greater amount of freedom allows the robot to reduce the cost, slightly in this case.

Consider the example environment shown in Fig. 3(c), an environment with obstacles. The start configuration is in the lower left corner, and the goal is on the far right. We assume collisions prevent the robot from moving forward, and this causes the Gaussian approximation of belief to be even less useful. In this environment, we evaluate policies not only based on μ_i and σ_i, but also on the probability of failure p_i^{fail} that corresponds to the goal not being achieved in 200 stages. We will discuss two cases with varying levels of process noise.

For both cases, the path through the graph chosen by

M_1 and M_2 is the one that weaves through the two narrow passages in the workspace. In the case with very little process noise, M_3 chooses the same nominal path. For this example we have $\mu_3, \sigma_3^2 = (1.984, 1.910)$, $p_3^{\text{fail}} = 0$, as opposed to $\mu_2, \sigma_2^2 = (0.868, 0.199)$ but with $p_2^{\text{fail}} = 0.225$. These results can be explained by M_3 attempting to move the robot between the obstacles, but when the policy becomes more uncertain of being able to navigate the passage it is able to take an alternate route. These alternate routes are more uncertain so the average uncertainty increases. However, M_2 fails frequently, but when it succeeds the results are good. One would expect similar mean and variance for M_3 on the sample paths that successfully navigate between the obstacles.

When we increase the amount of process noise, M_3 recognizes that it is frequently unable to navigate the first narrow passage, and attempting to do so typically causes the uncertainty to increase before taking an alternate route. Thus, M_3 nominally chooses the lower path around the first set of obstacles, then moves up to the middle route to take the second narrow passage, which is closer to the beacons. This results in $\mu_3, \sigma_3^2 = (0.848, 0.206)$, $p_3^{\text{fail}} = 0.088$, as opposed to $\mu_2, \sigma_2^2 = (0.881, 0.370)$, $p_2^{\text{fail}} = 0.203$. The penalty for the robot failing to reach its target can also be increased as a design decision, causing M_3 to nominally take the lower path through the entire graph to the goal.

VI. CONCLUSION

In this paper, we proposed a method to find minimum uncertainty paths for mobile robots using a POMDP value function approximation method. We discussed a general approach to generating belief-feedback policies using switched policies with fixed modes, and applied this method mobile robot navigation localizing with beacons. We demonstrated the performance of our method in simulation.

Our method excels, primarily, by considering observations besides the ML contingency, which required a parameterization of the belief that was more faithful to the true pdf. This allowed us to model nonlinear distortion to the belief, which is encountered in regions near beacons and around obstacles in the environment. Finally, the plan delivered is a true belief-feedback policy, meaning that the policy uses a different strategy from different beliefs, even if the projection of the two beliefs onto the configuration space is nearly identical.

Our proposed techniques can be generalized to find minimum uncertainty plans for robot systems with more complicated configuration spaces and non-Gaussian process and observation models, or to minimize cost objectives other than uncertainty. However, these systems may require other heuristics (rather than a workspace roadmap) to generate policies to explore the belief space. Developing these techniques for a general robot system is an interesting future direction for this work.

Although other techniques that optimize based on approximation techniques less faithful to the true POMDP may be faster, our method has demonstrated that it can generate a policy that improves system performance in a reasonable amount of time. As computing becomes faster, smaller, more parallel, and less expensive, this difference in planning times will become less significant in comparison to the performance gains that can be achieved on the physical robot system.

REFERENCES

[1] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *International Journal of Robotics Research*, 2009.

[2] R. He, S. Prentice, and N. Roy, "Planning in information space for a quadrotor helicopter in a gps-denied environment," in *IEEE International Conference on Robotics and Automation*, 2008.

[3] L. Mihaylova, J. De Schutter, and H. Bruyninckx, "A multisine approach for trajectory optimization based on information gain," *Robotics and Autonomous Systems*, vol. 43, no. 4, pp. 231–243, 2003.

[4] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99 – 134, 1998.

[5] J. Davidson and S. Hutchinson, "Hyper-particle filtering for stochastic systems," in *IEEE International Conference on Robotics and Automation*, 2008, pp. 2770–2777.

[6] S. Candido, J. Davidson, and S. Hutchinson, "Exploiting domain knowledge in planning for uncertain robot systems modeled as POMDPs," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2010, pp. 3596–3603.

[7] M. Apaydin, D. Brutlag, C. Guestrin, D. Hsu, J. Latombe, and C. Varm, "Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion," *Journal of Computational Biology*, vol. 10, pp. 257–281, 2003.

[8] R. Alterovitz, T. Simeon, and K. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty," in *Robotics: Science and Systems*, 2007.

[9] P. Missiuro and N. Roy, "Adapting probabilistic roadmaps to handle uncertain maps," in *IEEE International Conference on Robotics and Automation*, 2006, pp. 1261–1267.

[10] N. Melchior and R. Simmons, "Particle RRT for path planning with uncertainty," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1617–1624.

[11] J. Davidson and S. Hutchinson, "A sampling hyperbelief optimization technique for stochastic systems," in *Workshop on the Algorithmic Foundations of Robotics*, 2008.

[12] W. Burgard, D. Fox, and S. Thrun, "Active mobile robot localization by entropy minimization," in *Proceedings of the Second Euromicro Workshop on Advanced Mobile Robots*, 1997, pp. 155–162.

[13] N. Roy, W. Burgard, D. Fox, and S. Thrun, "Coastal navigation: Mobile robot navigation with uncertainty in dynamic environments," in *IEEE International Conference on Robotics and Automation*, 1999, pp. 35–40.

[14] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in linear pomdps by factoring the covariance," in *International Symposium on Robotics Research*, 2007.

[15] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[16] J. Porta, N. Vlassis, M. Spaan, and P. Poupart, "Point-based value iteration for continuous POMDPs," *The Journal of Machine Learning Research*, vol. 7, p. 2367, 2006.

[17] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[18] W. Lovejoy, "A survey of algorithmic methods for partially observed Markov decision processes," *Annals of Operations Research*, vol. 28, no. 1, pp. 47–65, 1991.

[19] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Second Edition*. Belmont, MA: Athena Scientific, 1995.

[20] S. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability, Second Edition*. New York, NY: Cambridge University Press, 2009.

[21] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer Verlag, 2001.

[22] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *IEEE International Conference on Robotics and Automation*, 1999, pp. 1322–1328.