Active Policy Learning for Robot Planning and Exploration under Uncertainty

Ruben Martinez-Cantin*, Nando de Freitas†, Arnaud Doucet†, José A. Castellanos*
* Department of Computer Science and Systems Engineering, University of Zaragoza
Email: {rmcantin,jacaste}@unizar.es
† Department of Computer Science, University of British Columbia
Email: {nando,arnaud}@cs.ubc.ca

***Abstract*— This paper proposes a simulation-based active policy learning algorithm for finite-horizon, partially-observed sequential decision processes. The algorithm is tested in the domain of robot navigation and exploration under uncertainty. In such a setting, the expected cost, that must be minimized, is a function of the belief state (filtering distribution). This filtering distribution is in turn nonlinear and subject to discontinuities, which arise because constraints in the robot motion and control models. As a result, the expected cost is non-differentiable and very expensive to simulate. The new algorithm overcomes the first difficulty and reduces the number of required simulations as follows. First, it assumes that we have carried out previous simulations which returned values of the expected cost for different corresponding policy parameters. Second, it fits a Gaussian process (GP) regression model to these values, so as to approximate the expected cost as a function of the policy parameters. Third, it uses the GP predicted mean and variance to construct a statistical measure that determines which policy parameters should be used in the next simulation. The process is then repeated using the new parameters and the newly gathered expected cost observation.**

Since the objective is to find the policy parameters that minimize the expected cost, this iterative active learning approach effectively trades-off between exploration (in regions where the GP variance is large) and exploitation (where the GP mean is low). In our experiments, a robot uses the proposed algorithm to plan an optimal path for accomplishing a series of tasks, while maximizing the information about its pose and map estimates. These estimates are obtained with a standard filter for simultaneous localization and mapping. Upon gathering new observations, the robot updates the state estimates and is able to replan a new path in the spirit of open-loop feedback control.

I. Introduction

The direct policy search method for reinforcement learning has led to significant achievements in control and robotics [1, 2, 3, 4]. The success of the method does often, however, hinge on our ability to formulate expressions for the gradient of the expected cost [5, 4, 6]. In some important applications in robotics, such as exploration, constraints in the robot motion and control models make it hard, and often impossible, to compute derivatives of the cost function with respect to the robot actions. In this paper, we present a direct policy search method for continuous policy spaces that relies on active learning to side-step the need for gradients.

The proposed active policy learning approach also seems to be more appropriate in situations where the cost function has many local minima that cause the gradient methods to get stuck. Moreover, in situations where the cost function is very expensive to evaluate by simulation, an active learning approach that is designed to minimize the number of evaluations might be more suitable than gradient methods, which often require small step sizes for stable convergence (and hence many cost evaluations).

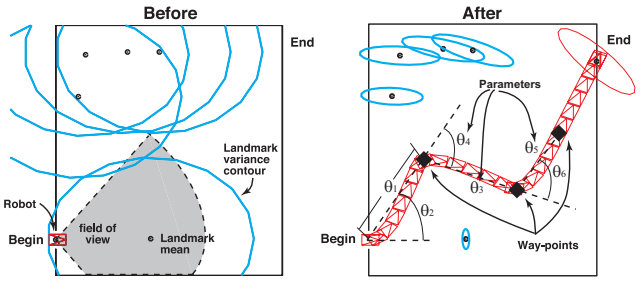

Fig. 1. The robot plans a path that allows it to accomplish the task of going from "Begin" to "End" while simultaneously reducing the uncertainty in the map and pose (robot location and heading) estimates. The robot has a prior over its pose and the landmark locations, but it can only see the landmarks within its field of view (left). In the planning stage, the robot must compute the best six-dimensional policy vector consisting of the three way-points describing the path. After running the stochastic planning algorithm proposed in this paper, the robot has reached the "End" while following a planned trajectory that minimizes the posterior uncertainty about its pose and map (right). The marginal landmark uncertainty ellipses are scaled for clarification.

We demonstrate the new approach on a hard robotics problem: planning and exploration under uncertainty. This problem plays a key role in simultaneous localization and mapping (SLAM), see for example [7, 8]. Mobile robots must maximize the size of the explored terrain, but, at the same time, they must ensure that localization errors are minimized. While exploration is needed to find new features, the robot must return to places were known landmarks are visible to maintain reasonable map and pose (robot location and heading) estimates.

In our setting, the robot is assumed to have a rough *a priori* estimate of the map features and its own pose. The robot must accomplish a series of tasks while simultaneously maximizing its information about the map and pose. This is illustrated in Figure 1, where a robot has to move from "Begin" to "End" by planning a path that satisfies logistic and physical constraints. The planned path must also result in improved map and pose estimates. *As soon as the robot accomplishes a task, it has a new a posteriori map that enables it to carry out future tasks in the same environment more efficiently.* This sequential decision making problem is exceptionally difficult because the actions

and states are continuous and high-dimensional. Moreover, the cost function is not differentiable and depends on the posterior belief (filtering distribution). Even a toy problem requires enormous computational effort. As a result, it is not surprising that most existing approaches relax the constrains. For instance, full observability is assumed in [9, 7], known robot location is assumed in [10], myopic planning is adopted in [8], and discretization of the state and/or actions spaces appears in [11, 12, 7]. *The method proposed in this paper does not rely on any of these assumptions.*

Our direct policy solution uses an any-time probabilistic active learning algorithm to predict what policies are likely to result in higher expected returns. The method effectively balances the goals of exploration and exploitation in policy search. It is motivated by work on experimental design [13, 14, 15]. Simpler variations of our ideas appeared early in the reinforcement literature. In [16], the problem is treated in the framework of exploration/exploitation with bandits. An extension to continuous spaces (infinite number of bandits) using locally weighted regression was proposed in [17]. Our paper presents richer criteria for active learning as well suitable optimization objectives.

This paper also presents posterior Cramér-Rao bounds to approximate the cost function in robot exploration. The appeal of these bounds is that they are much cheaper to simulate than the actual cost function.

Although the discussion is focused on robot exploration and planning, our policy search framework extends naturally to other domains. Related problems appear the fields of terrain-aided navigation [18, 9] and dynamic sensor nets [19, 6].

II. Application to Robot Exploration and Planning

Although the algorithm proposed in this paper applies to many sequential decision making settings, we will restrict attention to the robot exploration and planning domain. In this domain, the robot has to plan a path that will improve its knowledge of its pose (location and heading) and the location of navigation landmarks. In doing so, the robot might be subject to other constraints such as low energy consumption, limited time, safety measures and obstacle avoidance. However, for the time being, let us first focus on the problem of minimizing posterior errors in localization and mapping as this problem already captures a high degree of complexity.

There are many variations of this problem, but let us consider the one of Figure 1 for illustration purposes. Here, the robot has to navigate from "Begin" to "End" while improving its estimates of the map and pose. For the time being, let us assume that the robot has no problem in reaching the target. Instead, let us focus on how the robot should plan its path so as to improve its map and pose posterior estimates. Initially, as illustrated by the ellipses on the left plot, the robot has vague priors about its pose and the location of landmarks. We want the robot to plan a path (parameterized policy $\pi(\boldsymbol{\theta})$) so that by the time it reaches the target, it has learned the most about its pose and the map. This way, *if the robot has to repeat the task, it will have a better estimate of the map and hence it will be able to accomplish the task more efficiently.*

In this paper, the policy is simply a path parameterized as a set of ordered way-points $\boldsymbol{\theta}_i$, although different representations can be used depending on the robot capabilities. A trajectory with 3 way-points, whose location was obtained using our algorithm, is shown on the right plot of Figure 1. We use a standard proportional-integral-derivative (PID) controller to generate the motion commands $\mathbf{a} = \{\mathbf{a}_{1:T}\}$ to follow the path for T steps. The controller moves the robot toward each way-point in turn while taking into account the kinematic and dynamic constrains of the problem.

It should be noticed that the robot has a limited field of view. It can only see the landmarks that "appear" within an *observation gate*.

Having restricted the problem to one of improving posterior pose and map estimates, a natural cost function is the average mean square error (AMSE) of the state:

$$C^{\pi}_{AMSE} = \mathbb{E}_{p(\mathbf{x}_{0:T}, \mathbf{y}_{1:T}|\boldsymbol{\pi})} \left[\sum_{t=1}^{T} \lambda^{T-t} (\widehat{\mathbf{x}}_t - \mathbf{x}_t)(\widehat{\mathbf{x}}_t - \mathbf{x}_t)' \right],$$

where $\widehat{\mathbf{x}}_t = \mathbb{E}_{p(\mathbf{x}_t|\mathbf{y}_{1:t}, \boldsymbol{\pi})}[\mathbf{x}_t]$. The expectation is with respect to $p(\mathbf{x}_{0:T}, \mathbf{y}_{1:T}|\boldsymbol{\pi}) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{a}_t, \mathbf{x}_{t-1}) p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{a}_t)$, $\lambda \in [0, 1]$ is a discount factor, $\pi(\boldsymbol{\theta})$ denotes the policy parameterized by the way-points $\boldsymbol{\theta}_i \in \mathbb{R}^{n_\theta}$, $\mathbf{x}_t \in \mathbb{R}^{n_x}$ is the hidden state (robot pose and location of map features) at time t, $\mathbf{y}_{1:T} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T\} \in \mathbb{R}^{n_y T}$ is the history of observations along the planned trajectory for T steps, $\mathbf{a}_{1:T} \in \mathbb{R}^{n_a T}$ is the history of actions determined by the policy $\pi(\boldsymbol{\theta})$ and $\widehat{\mathbf{x}}_t$ is the posterior estimate of the state at time t.

In our application to robotics, we focus on the uncertainty of the posterior estimates at the end of the planning horizon. That is, we set λ so that the cost function reduces to:

$$C^{\pi}_{AMSE} = \mathbb{E}_{p(\mathbf{x}_T, \mathbf{y}_{1:T}|\boldsymbol{\pi})} \left[(\widehat{\mathbf{x}}_T - \mathbf{x}_T)(\widehat{\mathbf{x}}_T - \mathbf{x}_T)' \right], \quad (1)$$

Note that the true state $\mathbf{x}_T$ and observations are unknown in advance and so one has to marginalize over them.

The cost function hides an enormous degree of complexity. It is a matrix function of an *intractable filtering distribution* $p(\mathbf{x}_T|\mathbf{y}_{1:T}, \boldsymbol{\pi})$ (also known as the belief or information state). This belief can be described in terms of the observation and odometry (robot dynamics) models using marginalization and Bayes rule. The computation of this belief is known as the simultaneous localization and mapping problem (SLAM) and it is known to be notoriously hard because of nonlinearity and non-Gaussianity. Moreover, in our domain, the robot only sees the landmarks within and observation gate.

Since the models are not linear-Gaussian, one cannot use standard linear-quadratic-Gaussian (LQG) controllers [20] to solve our problem. Moreover, since the action and state spaces are large-dimensional and continuous, one cannot discretize the problem and use closed-loop control as suggested in [21]. That is, the discretized partially observed Markov decision process is too large for stochastic dynamic programming [22].

As a result of these considerations, we adopt the direct policy search method [23, 24]. In particular, the initial policy is set either randomly or using prior knowledge. Given this policy, we conduct simulations to estimate the AMSE. These simulations involve sampling states and observations using the prior, dynamic and observation models. They also involve estimating the posterior mean of the state with suboptimal filtering. After evaluating the AMSE using the simulated

1) Choose an initial policy π_0.
2) For $j = 1 : MaxNumberOfPolicySearchIterations$:
   a) For $i = 1 : N$:
      i) Sample the prior states $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$.
      ii) For $t = 1 : T$:
         A) Use a PID controller regulated about the path π_j to determine the current action $\mathbf{a}_t^{(i)}$.
         B) Sample the state $\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t|\mathbf{a}_t^{(i)}, \mathbf{x}_{t-1}^{(i)})$.
         C) Generate observations $\mathbf{y}_t^{(i)} \sim p(\mathbf{y}_t|\mathbf{a}_t^{(i)}, \mathbf{x}_t^{(i)})$ as described in Section II-A. There can be missing observations.
         D) Compute the filtering distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t}^{(i)}, \mathbf{a}_{1:t}^{(i)})$ using a SLAM filter.
   b) Evaluate the approximate AMSE cost function of equation (2) using the simulated trajectories.
   c) Use the active learning algorithm with Gaussian processes, described in Section III, to generate the new policy π_{j+1}. The choice of the new policy is governed by our desire to exploit and our need to explore the space of policies (navigation paths). In particular, we give preference to policies for which we expect the cost to be minimized and to policies where we have high uncertainty about what the cost might be.

Fig. 2. The overall solution approach in the open-loop control (OLC) setting. Here, N denotes the number of Monte Carlo samples and T is the planning horizon. In replanning with open-loop feedback control (OLFC), one simply uses the present position and the estimated posterior distribution (instead of the prior) as the starting point for the simulations. One can apply this strategy with either approaching or receding control horizons. It is implicit in the pseudo-code that we freeze the random seed generator so as to reduce variance.

trajectories, we update the policy parameters and iterate with the goal of minimizing the AMSE. Note that in order to reduce Monte Carlo variance, the random seed should be frozen as described in [24]. The pseudo-code for this open-loop simulation-based controller (OLC) is shown in Figure 2.

Note that as the robot moves along the planned path, it is possible to use the newly gathered observations to update the posterior distribution of the state. This distribution can then be used as the prior for subsequent simulations. This process of replanning is known as open-loop feedback control (OLFC) [20]. We can also allow for the planning horizon to recede. That is, as the robot moves, it keeps planning T steps ahead of its current position. This control framework is also known as receding-horizon model-predictive control [25].

In the following two subsections, we will describe a way of conducting the simulations to estimate the AMSE. The active policy update algorithm will be described in Section III.

A. Simulation of the cost function

We can approximate the AMSE cost by simulating N state and observation trajectories $\{\mathbf{x}_{1:T}^{(i)}, \mathbf{y}_{1:T}^{(i)}\}_{i=1}^N$ and adopting the Monte Carlo estimator:

$$C_{AMSE}^{\pi} \approx \frac{1}{N} \sum_{i=1}^{N} (\widehat{\mathbf{x}}_T^{(i)} - \mathbf{x}_T^{(i)})(\widehat{\mathbf{x}}_T^{(i)} - \mathbf{x}_T^{(i)})'. \qquad (2)$$

Assuming that π is given (we discuss the active learning algorithm to learn π in Section III), one uses a PID controller to obtain the next action $\mathbf{a}_t$. The new state $\mathbf{x}_t$ is easily simulated using the odometry model. The process of generating observations is more involved. As shown in Figure 3, for each landmark, one draws a sample from its posterior. If the sample falls within the observation gate, it is treated as an observation. As in most realistic settings, most landmarks will remain unobserved.

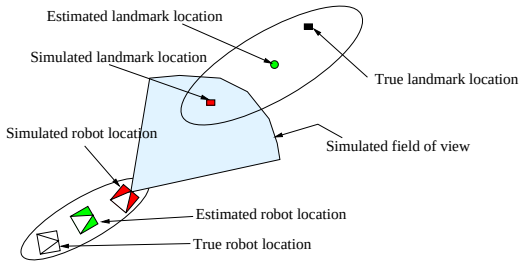

Fig. 3. An observation is generated using the current map and robot pose estimates. Gating information is used to validate the observation. In this picture, the simulation validates the observation despite the fact that the true robot and feature locations are too distant for the given field of view. New information is essential to reduce the uncertainty and improve the simulations.

After the trajectories $\{\mathbf{x}_{1:T}^{(i)}, \mathbf{y}_{1:T}^{(i)}\}_{i=1}^N$ are obtained, one uses a SLAM filter (EKF, UKF or particle filter) to compute the posterior mean state $\widehat{\mathbf{x}}_{1:T}^{(i)}$. (In this paper, we adopt the EKF-SLAM algorithm to estimate the mean and covariance of this distribution. We refer the reader to [26] for implementation details.) *The evaluation of the cost function is therefore extremely expensive.* Moreover, since the model is nonlinear, it is hard to quantify the uncertainty introduced by the suboptimal filter. Later, in Section IV, we will discuss an alternative cost function, which consists of a lower bound on the AMSE. Yet, in both cases, it is imperative to minimize the number of evaluations of the cost functions. This calls for an active learning approach.

III. Active Policy Learning

This section presents an active learning algorithm to update the policy parameters after each simulation. In particular, we adopt the expected cost simulation strategy presented in [24]. In this approach, a scenario consists of an initial choice of the state and a sequence of random numbers. Given a policy parameter vector and a set of fixed scenarios, the simulation is deterministic and yields an empirical estimate of the expected cost [24].

The simulations are typically very expensive and consequently cannot be undertaken for many values of the policy parameters. Discretization of the potentially high-dimensional and continuous policy space is out of the question. The standard solution to this problem is to optimize the policy using gradients. However, the local nature of gradient-based optimization often leads to the common criticism that direct policy search methods "get stuck" in local minima. Even more pertinent to our setting, is the fact that the cost function is discontinuous and hence policy gradient methods do not apply. We present an alternative approach to gradient-based optimization for continuous policy spaces. This approach, which we refer to as active policy learning, is based on experimental design ideas [27, 13, 28, 29]. Active policy learning is an any-time, "black-box" statistical optimization

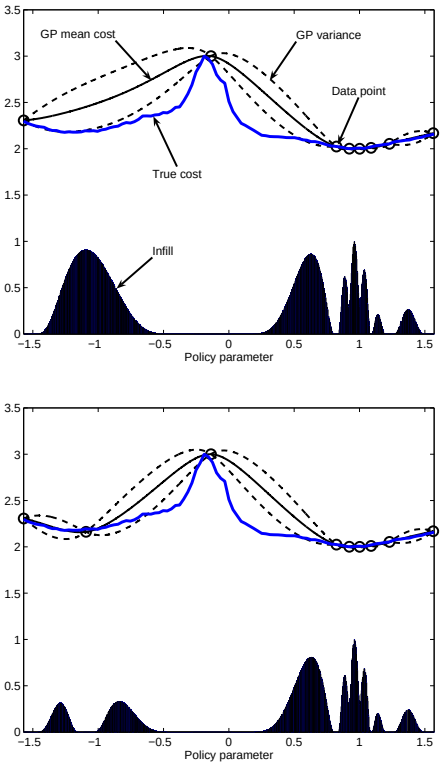

Fig. 4. An example of active policy learning with a univariate policy using data generated by our simulator. The figure on top shows a GP approximation of the cost function using 8 simulated values. The variance of the GP is low, but not zero, at these simulated values. In reality, the true expected cost function is unknown. The figure also shows the expected improvement (infill) of each potential next sampling location in the lower shaded plot. The infill is high where the GP predicts a low expected cost (exploitation) and where the prediction uncertainty is high (exploration). Selecting and labelling the point suggested by the highest infill in the top plot produces the GP fit in the plot shown below. The new infill function, in the plot below, suggests that we should query a point where the cost is expected to be low (exploitation).

approach. Figure 4 illustrates it for a simple one-dimensional example. The approach is iterative and involves three steps.

In the first step, a Bayesian regression model is learned to map the policy parameters to the estimates of the expected cost function obtained from previous simulations. In this work, the regression function is obtained using Gaussian processes (GPs). Though in Figure 4 the GPs provide a good approximation to the expected cost, it should be emphasized that the objective is not to predict the value of the regression surface over the entire feasible domain, but rather to predict it well near the minima. The details of the GP fit are presented in Section III-A.

The second step involves active learning. Because the simulations are expensive, we must ensure that the selected samples (policy parameter candidates) will generate the maximum possible improvement. Roughly speaking, it is reasonable to sample where the GP predicts a low expected cost (exploitation) or where the GP variance is large (exploration). These intuitions can be incorporated in the design of a statistical measure indicating where to sample. This measure is known as the infill function, borrowing the term from the geostatistics literature. Figure 4 depicts a simple infill function that captures our intuitions. More details on how to choose the infill are presented in Section III-B.

Having defined an infill function, still leaves us with the problem of optimizing it. This is the third and final step in the approach. Our thesis is that the infill optimization problem is more amenable than the original problem because in this case the cost function is known and easy to evaluate. Furthermore, for the purposes of our application, it is not necessary to guarantee that we find the global minimum, merely that we can quickly locate a point that is likely to be as good as possible.

To deal with this nonlinear constrained optimization problem, we adopted the DIvided RECTangles (DIRECT) algorithm [30, 31]. DIRECT is a deterministic, derivative-free sampling algorithm. It uses the existing samples of the objective function to decide how to proceed to divide the feasible space into finer rectangles. For low-dimensional parameter spaces, say up to 10D, DIRECT provides a better solution than gradient approaches because the infill function tends to have many local optima. Another motivating factor is that DIRECT's implementation is easily available [32]. However, we conjecture that for large dimensional spaces, sequential quadratic programming or concave-convex programming [33] might be better algorithm choices for infill optimization.

A. Gaussian processes

A *Gaussian process*, $\mathbf{z}(\cdot) \sim GP(m(\cdot), K(\cdot,\cdot))$, is an infinite random process indexed by the vector $\boldsymbol{\theta}$, such that any realization $\mathbf{z}(\boldsymbol{\theta})$ is Gaussian [34]. We can parameterize the GP hierarchically

$$\begin{aligned} C^{\pi}(\boldsymbol{\theta}) &= \mathbf{1}\mu + \mathbf{z}(\boldsymbol{\theta}) \\ \mathbf{z}(\cdot) &\sim GP(0, \sigma^2 K(\cdot,\cdot)) \end{aligned}$$

and subsequently estimate the posterior distributions of the mean μ and scale σ^2 using standard Bayesian conjugate analysis, see for example [14]. The symbol $\mathbf{1}$ denotes a column vector of ones. Assuming that n simulations have been conducted, the simulated costs $\{C^{\pi}_{1:n}\}$ and the predicted cost C^{π}_{n+1} for a new test point $\boldsymbol{\theta}_{n+1}$ are jointly Gaussian:

$$\begin{bmatrix} C^{\pi}_{n+1} \\ C^{\pi}_{1:n} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 1 \\ \mathbf{1} \end{bmatrix} \mu, \sigma^2 \begin{bmatrix} k & \mathbf{k}^T \\ \mathbf{k} & \mathbf{K} \end{bmatrix} \right),$$

where $\mathbf{k}^T = [k(\boldsymbol{\theta}_{n+1}, \boldsymbol{\theta}_1) \cdots k(\boldsymbol{\theta}_{n+1}, \boldsymbol{\theta}_n)]$, $k = k(\boldsymbol{\theta}_{n+1}, \boldsymbol{\theta}_{n+1})$ and $\mathbf{K}$ is the training data kernel matrix with entries $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, n$. Since we are interested in regression, the Matern kernel is a suitable choice for $k(\cdot|\cdot)$ [14].

We assign a normal-inverse-Gamma conjugate prior to the parameters: $\mu \sim \mathcal{N}(0, \sigma^2\delta^2)$ and $\sigma^2 \sim \mathcal{IG}(a/2, b/2)$. The priors play an essential role at the beginning of the design process, when there are only a few data. Classical Bayesian analysis allow us to obtain analytical expressions for the posterior modes of these quantities:

$$\begin{aligned} \widehat{\mu} &= (\mathbf{1}^T\mathbf{K}^{-1}\mathbf{1} + \delta^{-2})^{-1}\mathbf{1}^T\mathbf{K}^{-1}C^{\pi} \\ \widehat{\sigma}^2 &= \frac{b + C^{\pi T}\mathbf{K}^{-1}C^{\pi} - (\mathbf{1}^T\mathbf{K}^{-1}\mathbf{1} + \delta^{-2})\widehat{\mu}^2}{n + a + 2} \end{aligned}$$

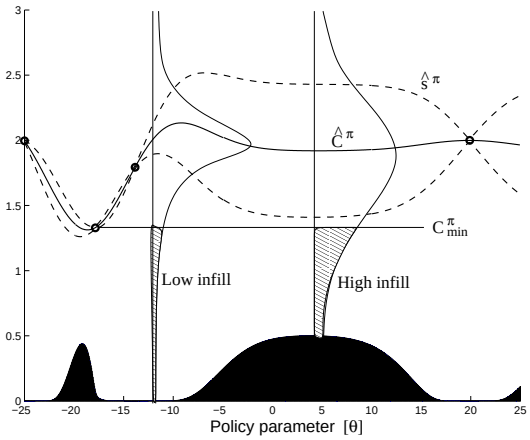

Fig. 5. Infill function.

Using the previous estimates, the GP predictive mean and variance are given by

$$\begin{aligned}\widehat{C}^{\pi}(\boldsymbol{\theta}) &= \widehat{\mu} + \mathbf{k}^T\mathbf{K}^{-1}(C^{\pi}_{1:n} - \mathbf{1}\widehat{\mu}) \\ \widehat{s}^2(\boldsymbol{\theta}) &= \widehat{\sigma}^2\left\{k - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k} + \frac{(1-\mathbf{1}^T\mathbf{K}^{-1}\mathbf{k})^2}{(\mathbf{1}^T\mathbf{K}^{-1}\mathbf{1} + \delta^{-2})}\right\}.\end{aligned}$$

Since the number of query points is small, the GP predictions are very easy to compute.

B. Infill Function

Let $C^{\pi}_{\min}$ denote the current lowest (best) estimate of the cost function. As shown in Figure 5, we can define the probability of improvement at a point $\boldsymbol{\theta}$ to be

$$p(C^{\pi}(\boldsymbol{\theta}) \leq C^{\pi}_{\min}) = \Phi\left(\frac{C^{\pi}_{\min} - \widehat{C}^{\pi}(\boldsymbol{\theta})}{\widehat{s}(\boldsymbol{\theta})}\right),$$

where $C^{\pi}(\boldsymbol{\theta}) \sim \mathcal{N}(\widehat{C}^{\pi}(\boldsymbol{\theta}), \widehat{s}(\boldsymbol{\theta})^2)$ and Φ denotes CDF of the standard Normal distribution. This measure was proposed several decades ago by [27], who used univariate Wiener process. However, as argued by [13], it is sensitive to the value of $C^{\pi}_{\min}$. To overcome this problem, Jones defined the improvement over the current best point as $I(\boldsymbol{\theta}) = \max\{0, C^{\pi}_{\min} - C^{\pi}(\boldsymbol{\theta})\}$. This resulted in the following expected improvement (infill function):

$$EI(\boldsymbol{\theta}) = \begin{cases} (C^{\pi}_{\min} - \widehat{C}^{\pi}(\boldsymbol{\theta}))\Phi(d) + \widehat{s}(\boldsymbol{\theta})\phi(d) & \text{if } \widehat{s} > 0 \\ 0 & \text{if } \widehat{s} = 0 \end{cases}$$

where ϕ is the PDF of the standard Normal distribution and

$$d = \frac{C^{\pi}_{\min} - \widehat{C}^{\pi}(\boldsymbol{\theta})}{\widehat{s}(\boldsymbol{\theta})}.$$

IV. A Cheaper Cost: The Posterior Cramér-Rao Bound

As mentioned in Section II-A, it is not possible to compute the AMSE cost function exactly. In that section, we proposed a simulation approach that required that we run an SLAM filter for each simulated scenario. This approximate filtering step is not only expensive, but also a possible source of errors when approximating the AMSE with Monte Carlo simulations.

The posterior Cramér-Rao bound (PCRB) for nonlinear systems leads to an alternative objective function that is cheaper to evaluate and *does not require that we run a SLAM filter*. That is, the criterion presented next does not require the adoption of an EKF, UKF, particle filter or any other suboptimal filter in order to evaluate it. The PCRB is a "measure" of the maximum information that can be extracted from the dynamic system when both the measurements and states are assumed random. It is defined as the inverse of the Fisher information matrix $\mathbf{J}$ and provides the following lower bound on the AMSE:

$$C^{\pi}_{AMSE} \geq C^{\pi}_{PCRB} = \mathbf{J}^{-1}$$

Tichavský [35], derived the following Riccati-like recursion to compute the PCRB for any unbiased estimator:

$$\mathbf{J}_{t+1} = \mathbf{D}_t - \mathbf{C}_t'(\mathbf{J}_t + \mathbf{B}_t)^{-1}\mathbf{C}_t + \mathbf{A}_{t+1}, \tag{3}$$

where

$$\begin{aligned}\mathbf{A}_{t+1} &= \mathbb{E}[-\Delta_{x_{t+1},x_{t+1}} \log p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1})] \\ \mathbf{B}_t &= \mathbb{E}[-\Delta_{x_t,x_t} \log p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{a}_t)] \\ \mathbf{C}_t &= \mathbb{E}[-\Delta_{x_t,x_{t+1}} \log p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{a}_t)] \\ \mathbf{D}_t &= \mathbb{E}[-\Delta_{x_{t+1},x_{t+1}} \log p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{a}_t)],\end{aligned}$$

where the expectations are with respect to the simulated trajectories and Δ denotes the Laplacian operator. By simulating (sampling) trajectories, using our observation and transition models, one can easily approximate these expectations with Monte Carlo averages. *These averages can be computed off-line and hence the expensive recursion of equation (3) only needs to be done once for all scenarios.*

The PCRB approximation method of [35] applies to nonlinear (NL) models with additive noise only. This is not the case in our setting and hence a potential source of error. An alternative PCRB approximation method that overcomes this shortcoming, in the context of jump Markov linear (JML) models, was proposed by [36]. We try both approximations in our experiments and refer to them as NL-PCRB and JML-PCRB respectively.

The AMSE simulation approach of Section II-A using the EKF requires that we perform an expensive Ricatti update (EKF covariance update) for each simulated trajectory. In contrast, the simulation approach using the PCRB only requires one Ricatti update (equation (3)). Thus, the latter approach is considerably cheaper. Yet, the PCRB is only a lower bound and hence it is not guaranteed to be necessarily tight. In the following section, we will provide empirical comparisons between these simulation approaches.

V. Experiments

We present two sets of experiments. The first experiment is very simple as it is aimed at illustrating the approach. It involves a fixed-horizon stochastic planning domain. The second set of experiments is concerned with exploration with receding horizon policies in more realistic settings. In all cases, the aim is to find the optimal path in terms of posterior information about the map and robot pose. For clarification, other terms contributing to the cost, such as time and obstacles are not considered, but the implementation should be straightforward.

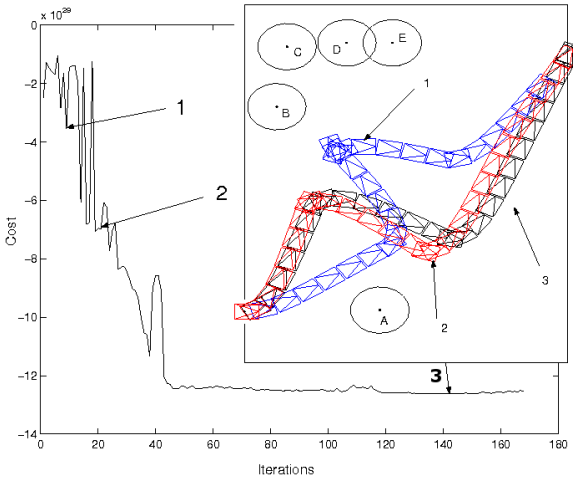

Fig. 6. Empirical AMSE cost as a function of policy improvement iterations. For this 6D parameter space, the solution converges to the minimum in few iterations, while allowing for several exploration steps in the first 40 iterations. The figure also shows the actual computed trajectories at three different iteration steps.

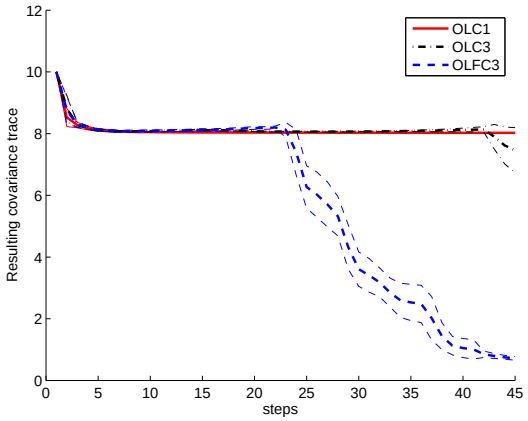

Fig. 8. Evolution of the trace of the state covariance matrix for 15 runs using OLC1, OLC3 and OLFC3, with 99% confidence intervals, for the map in Figure 7.

A. *Fixed-horizon planning*

The first experiment is the one described in Figure 1. Here, the start and end positions of the path are fixed. The robot has to compute the coordinates of three intermediate way-points and, hence, the policy has six parameters. For illustration purposes we chose a simple environment consisting of 5 landmarks (with vague priors). We placed an informative prior on the initial robot pose. Figure 6 shows three different robot trajectories computed during policy optimization. The trajectories are also indicated in the Monte Carlo AMSE cost evolution plot. The 6D optimization requires less than 50 iterations. We found that the optimal trajectory allowed the robot to observe the maximum number of features. However, since the prior on the robot's initial pose is informative (narrow Gaussian), feature A is originally detected with very low uncertainty. Consequently, the robot tries to maintain that feature in the field of view to improve the localization. A greedy strategy would have focused only on feature A, improving the estimation of that feature and the robot, but dropping the global posterior estimate.

B. *Receding-horizon planning*

In this experiment, the a priori map has high uncertainty (1 meter standard deviation – see Figure 7). The robot is a differential drive vehicle equipped with odometers and a stereo camera that provides the location of features. The field of view is limited to 7 meters and 90^o, which are typical values for reliable stereo matching. We assume that the camera and a detection system that provides a set of observations every 0.5 seconds. The sensor noise is Gaussian for both range and bearing, with standard deviations $\sigma_{range} = 0.2 \cdot range$ and $\sigma_{bearing} = 0.5^o$. The policy is given by a set of ordered way-points. Each way-point is defined in terms of heading and distance with respect to the robot pose at the preceding way-point. The distance between way-points is limited to 10 meters and the heading should be in the interval $[-3\pi/4, 3\pi/4]$ to avoid backwards trajectories. The motion commands are computed by a PID controller to guarantee that the goal is reached in 10 seconds.

First, we compare the behavior of the robot using different planning and acting horizons. The three methods that we implemented are:

- OLC1 : This is an open loop greedy algorithm that plans with only 1 way-point ahead. Once the way-point is reached, the robot replans a path. Strictly speaking this is a hybrid of OLC and OLFC as replanning using actual observations takes place. For simplicity, however, we refer to it as OLC.
- OLC3 : This is an open loop algorithm that plans with 3 way-points ahead. Once the third way-point is reached, the robot replans using actual observations.
- OLFC3 This is an open loop feedback controller with a receding horizon. The planning horizon is 3 way-points, but the execution horizon is only 1 step. Thus, the last 2 way-points plus a new way-point are recomputed after a way-point is reached.

It is obvious that the OLC algorithms have a lower computational cost. Using the AMSE cost and the map of Figure 7, the times for OLC1, OLC3 and OLFC3 are approximately 6, 30 and 75 minutes (using an un-optimized Matlab implementation). On the other hand, the OLC methods can get trapped in local minima, as shown in Figure 7. Due to the limited planning horizon of OLC1, it barely explores new areas. OLC3 tends to overshoot as it only replans at the third way-point. OLFC3, on the other hand, replans at each step and as a result is able to steer to the unexplored part of the map. Figure 8 plots the evolution of the uncertainty in this trap situation for 15 experimental runs. The controller with feedback is clearly the winner because it avoids the trap. This behavior is stable across different runs.

We repeated this experiment with different initial random maps (5 landmarks). Figure 9 shows the methods perform similarly as worst-case situations are rare.

The next set of simulations is used to experimentally

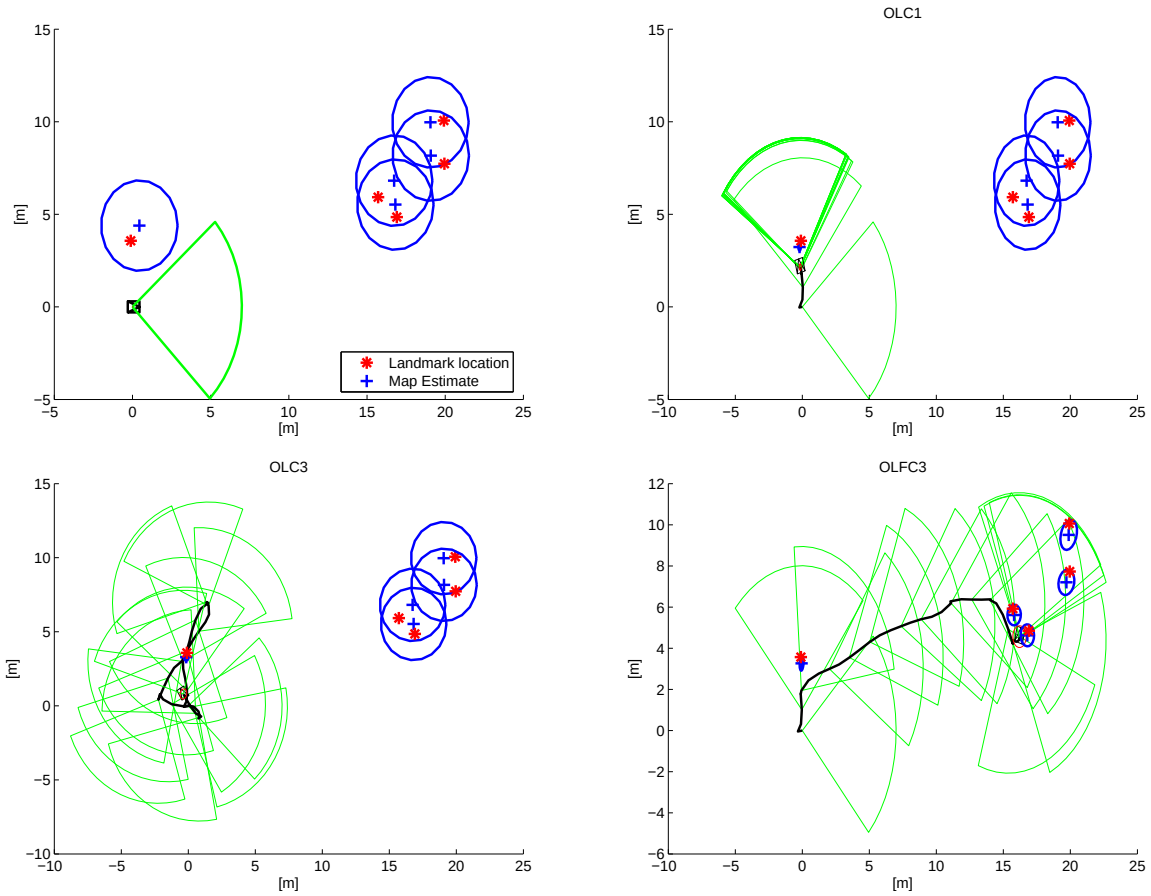

Fig. 7. Trajectories generated using OLC1, OLC3 and OLFC3. The blue and red ellipses represent the landmark and robot location 95% confidence intervals. The robot field of view is shown in green. OLC3 is more exploratory than OLC1, which "gets stuck" repeatedly updating the first landmark it encounters. Yet, only OLFC3, because of being able to replan at each step, is able to fully explore the map and reduce the uncertainty.

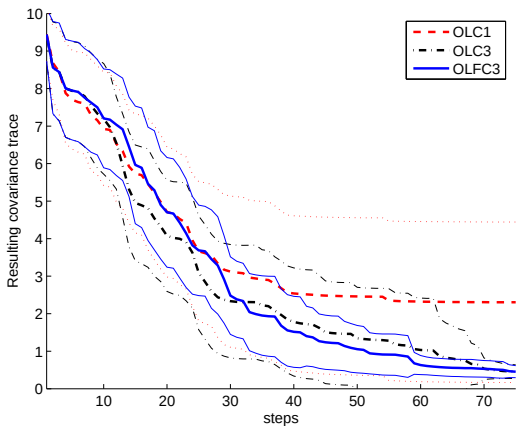

Fig. 9. Evolution of the trace of the state covariance matrix for 15 random maps using OLC1, OLC3 and OLFC3, with 99% confidence intervals.

validate the PCRB approximation of the AMSE cost. We increase the size and the complexity of the environment to 30 landmarks in a 25 by 25 meters squared area. Figure 10 shows the trace of the covariance matrix of the map and robot location, estimated using OLFC3, for the three approximations discussed in this paper. The JML-PCRB remains close to the simulated AMSE. This indicates that this bound is tight and a good choice in this case. On the other hand, NL-PCRB seems to be too loose. In this larger map, the computational times for the approximate AMSE and JML-PCRB were approximately 180 and 150 minutes respectively.

VI. Discussion and Future Work

We have presented an approach for stochastic exploration and planning rooted in strong statistical and decision-theoretic foundations. The most important next step is to test the proposed simulator on a real robotic domain. One needed step in this implementation is enabling the robot to add new landmarks to the existing map, while still within our decision-theoretic framework. We also note that our method is directly applicable to the problem of planning the architecture of a dynamic sensor network. In terms of modelling, we need to

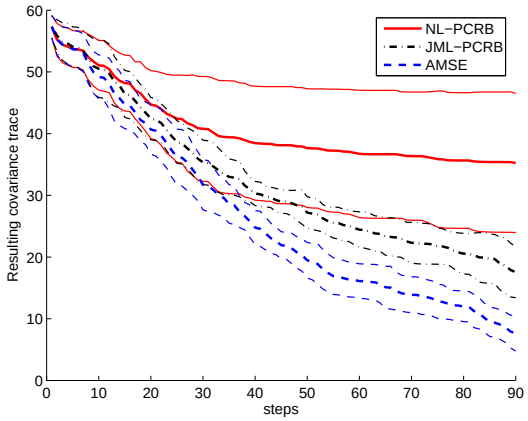

Fig. 10. Evolution of the trace of the state covariance matrix for 10 random maps with the AMSE, NL-PCRB and JML-PCRB cost functions, while using OLFC3.

introduce richer cost functions and constraints. In terms of algorithm improvement, we must design infill optimization strategies for high-dimensional policies. Whenever gradients are available, the approach presented here could be improved by ensuring that the regression function matches the gradients at the query points. Finally, on the theoretical front, we plan to build upon early work on correlated bandits to obtain theoretical performance bounds and, hence, confidence intervals that could potentially do better than the current infill criterion.

ACKNOWLEDGMENTS

We would like to thank Eric Brochu, Nick Roy, Hendrik Kueck and Andrew Ng for valuable discussions. This project was supported by NSERC, MITACS and the Dirección General de Investigación of Spain, project DPI2006-13578.

REFERENCES

[1] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *IEEE International Conference on Robotics and Automation*, 2004.

[2] G. Lawrence, N. Cowan, and S. Russell, "Efficient gradient estimation for motor control learning," in *Uncertainty in Artificial Intelligence*. San Francisco, CA: Morgan Kaufmann, 2003, pp. 354–36.

[3] A. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang, "Inverted autonomous helicopter flight via reinforcement learning," in *International Symposium on Experimental Robotics*, 2004.

[4] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *IEEE International Conference on Intelligent Robotics Systems*, 2006.

[5] J. Baxter and P. L. Bartlett, "Infinite-horizon policy-gradient estimation," *Journal of Artificial Intelligence Research*, vol. 15, pp. 319–350, 2001.

[6] S. Singh, N. Kantas, A. Doucet, B. N. Vo, and R. J. Evans, "Simulation-based optimal sensor scheduling with application to observer trajectory planning," in *IEEE Conference on Decision and Control and European Control Conference*, 2005, pp. 7296– 7301.

[7] R. Sim and N. Roy, "Global A-optimal robot exploration in SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, 2005.

[8] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using Rao-Blackwellized particle filters," in *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[9] S. Paris and J. P. Le Cadre, "Planification for terrain-aided navigation," in *Fusion 2002*, Annapolis, Maryland, July 2002, pp. 1007–1014.

[10] C. Leung, S. Huang, G. Dissanayake, and T. Forukawa, "Trajectory planning for multiple robots in bearing-only target localisation," in *IEEE International Conference on Intelligent Robotics Systems*, 2005.

[11] M. L. Hernandez, "Optimal sensor trajectories in bearings-only tracking," in *Proceedings of the Seventh International Conference on Information Fusion*, P. Svensson and J. Schubert, Eds., vol. II. Mountain View, CA: International Society of Information Fusion, Jun 2004, pp. 893–900.

[12] T. Kollar and N. Roy, "Using reinforcement learning to improve exploration trajectories for error minimization," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[13] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[14] T. J. Santner, B. Williams, and W. Notz, *The Design and Analysis of Computer Experiments*. Springer-Verlag, 2003.

[15] E. S. Siah, M. Sasena, and J. L. Volakis, "Fast parameter optimization of large-scale electromagnetic objects using DIRECT with Kriging metamodeling," *IEEE Transactions on Microwave Theory and Techniques*, vol. 52, no. 1, pp. 276–285, January 2004.

[16] L. P. Kaelbling, "Learning in embedded systems," Ph.D. dissertation, Stanford University, 1990.

[17] A. W. Moore and J. Schneider, "Memory-based stochastic optimization," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds., vol. 8. The MIT Press, 1996, pp. 1066–1072.

[18] N. Bergman and P. Tichavský, "Two Cramér-Rao bounds for terrain-aided navigation," *IEEE Transactions on Aerospace and Electronic Systems*, 1999, in review.

[19] M. L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom, "Multisensor resource deployment using posterior Cramèr-Rao bounds," *IEEE Transactions on Aerospace and Electronic Systems Aerospace*, vol. 40, no. 2, pp. 399– 416, April 2004.

[20] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[21] O. Tremois and J. P. LeCadre, "Optimal observer trajectory in bearings-only tracking for manoeuvring sources," *IEE Proceedings on Radar, Sonar and Navigation*, vol. 146, no. 1, pp. 31–39, 1999.

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

[23] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3, pp. 229–256, 1992.

[24] A. Y. Ng and M. I. Jordan, "Pegasus: A policy search method for large MDPs and POMDPs." in *Uncertainty in Artificial Intelligence (UAI2000)*, 2000.

[25] J. M. Maciejowski, *Predictive control: with constraints*. Prentice-Hall, 2002.

[26] J. A. Castellanos and J. D. Tardós, *Mobile Robot Localization and Map Building. A Multisensor Fusion Approach*. Kluwer Academic Publishers, 1999.

[27] H. J. Kushner, "A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise," *Journal of Basic Engineering*, vol. 86, pp. 97–106, 1964.

[28] D. R. Jones, "A taxonomy of global optimization methods based on response surfaces," *Journal of Global Optimization*, vol. 21, pp. 345–383, 2001.

[29] M. J. Sasena, "Flexibility and efficiency enhancement for constrained global design optimization with Kriging approximations," Ph.D. dissertation, University of Michigan, 2002.

[30] D. R. Jones, C. D. Perttunen, and B. E. Stuckman, "Lipschitzian optimization without the Lipschitz constant," *Journal of Optimization Theory and Applications*, vol. 79, no. 1, pp. 157–181, October 1993.

[31] J. M. Gablonsky, "Modification of the DIRECT algorithm," Ph.D. dissertation, Department of Mathematics, North Carolina State University, Raleigh, North Carolina, 2001.

[32] D. E. Finkel, *DIRECT Optimization Algorithm User Guide*. Center for Research in Scientific Computation, North Carolina State University, 2003.

[33] A. J. Smola, S. V. N. Vishwanathan, and T. Hofmann, "Kernel methods for missing variables," in *AI Stats*, 2005, pp. 325–332.

[34] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts: MIT Press, 2006.

[35] P. Tichavský, C. Muravchik, and A. Nehorai, "Posterior Cramér-Rao bounds for discrete-time nonlinear filtering," *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, 1998.

[36] N. Bergman, A. Doucet, and N. J. Gordon, "Optimal estimation and Cramér-Rao bounds for partial non-Gaussian state space models," *Annals of the Institute of Mathematical Statistics*, vol. 52, no. 1, pp. 1–17, 2001.