SACBP: Belief Space Planning for Continuous-Time Dynamical Systems via Stochastic Sequential Action Control

Haruki Nishimura and Mac Schwager

Stanford University, Stanford, CA 94305, USA
{hnishimura, schwager}@stanford.edu

Abstract. We propose a novel belief space planning technique for continuous dynamics by viewing the belief system as a hybrid dynamical system with time-driven switching. Our approach is based on the perturbation theory of differential equations and extends Sequential Action Control [1] to stochastic belief dynamics. The resulting algorithm, which we name SACBP, does not require discretization of spaces or time and synthesizes control signals in near real-time. SACBP is an anytime algorithm that can handle general parametric Bayesian filters under certain assumptions. We demonstrate the effectiveness of our approach in an active sensing scenario and a model-based Bayesian reinforcement learning problem. In these challenging problems, we show that the algorithm significantly outperforms other existing solution techniques including approximate dynamic programming and local trajectory optimization.

Keywords: Mobile Robots, Optimization and Optimal Control, Probabilistic Reasoning, Vision and Sensor-based Control

1 Introduction

Planning under uncertainty still remains as a challenge for robotic systems. Various types of uncertainty, including unmodeled dynamics, stochastic disturbances, and imperfect sensing, significantly complicate problems that are otherwise easy. For example, suppose that a robot needs to move an object from some initial state to a desired goal. If the mass properties of the object are not known beforehand, the robot needs to simultaneously estimate these parameters and perform control, while taking into account the effects of their uncertainty; the exploration and exploitation trade-off needs to be resolved [2]. On the other hand, uncertainty is quite fundamental in motivating some problems. For instance, a noisy sensor may encourage the robot to carefully plan a trajectory so the observations taken along it are sufficiently informative. This type of problems concerns pure information gathering and is often referred to as active sensing [3].

A principled approach to address all those problems is to form plans in the belief space, where the planner chooses sequential control inputs based on the evolution of the belief state. This approach enables the robot to appropriately execute controls under stochasticity and partial observability since they are both incorporated into the belief state. Belief space planning is also well suited for generating information gathering actions [4].

This paper proposes a novel online belief space planning algorithm. It does not require discretization of the state space or the action space, and can directly handle continuous-time system dynamics. The algorithm optimizes the expected value of the first-order cost reduction with respect to a nominal control policy at every re-planning time, proceeding in a receding horizon fashion. We are inspired by the Sequential Action Control (SAC) [1] algorithm recently proposed by Ansari and Murphey for model-based deterministic optimal control problems. SAC is an online method to synthesize control signals in real time for challenging (but deterministic) physical systems such as a cart pendulum and a spring-loaded inverted pendulum. Based on the concept of SAC, this paper develops an algorithmic framework to control stochastic belief systems whose dynamics are governed by parametric Bayesian filters.

1.1 Related Work in Belief Space Planning

Greedy Strategies Belief space planning is known to be challenging for a few reasons. First, the belief state is continuous and can be high-dimensional even if the underlying state space is small or discrete. Second, the dynamics that govern the belief state transitions are stochastic due to unknown future observations. Greedy approaches alleviate the complexity by ignoring long-term effects and solve single-shot decision making problems. Despite their suboptimality for long-term planning, these methods are often employed to find computationally tractable solutions and achieve reasonable performance in different problems [5–7], especially in the active sensing domain.

Trajectory Optimization Methods In contrast to the greedy approaches, trajectory optimization methods take into account multiple timesteps at once and find non-myopic solutions. In doing so, it is often assumed that the maximum likelihood observation (MLO) will always occur at the planning phase [4, 8, 9]. This heuristic assumption results in a deterministic optimal control problem, whereby various nonlinear trajectory optimization algorithms are applicable. However, ignoring the effects of stochastic future observations can degrade the performance [10]. Other methods [10, 11] that do not rely on the MLO assumption are advantageous in that regard. In particular, belief iLQG [10] performs iterative local optimization in the Gaussian belief space by quadratically approximating the value function and linearizing the dynamics to obtain a time-variant linear feedback policy. However, this method as well as many other solution techniques in this category result in multiple iterations of intensive computation and require significant amount of time until convergence.

Belief MDP and POMDP Approaches Belief space planning can be modeled as a Markov decision process (MDP) in the belief space, given that the belief state transition is Markovian. If in addition the reward (or cost) is defined as an explicit function of the state and the control, the problem is equivalent to a partially observable Markov decision process (POMDP) [12]. A key challenge in POMDPs and belief MDPs has been to address problems with large state spaces. This is particularly important in belief MDPs since the state space for a belief MDP is a continuous belief space. To handle continuous spaces, Couëtoux et al. [13] introduce double progressive widening (DPW) for Monte Carlo Tree Search (MCTS) [14]. In [2], this MCTS-DPW algorithm is run in the belief space to solve the object manipulation problem mentioned in Section 1. We have also presented a motion-based communication algorithm in our prior work [15], which uses MCTS-DPW for active intent inference with monocular vision.

While MCTS-DPW as well as other general purpose POMDP methods [16, 17] are capable of handling continuous state spaces, their algorithmic concepts are rooted in dynamic programming and tree search, requiring a sufficient amount of exploration in the tree. The tree search technique also implicitly assumes discrete-time transition models. In fact, most prior works discussed above are intended for discrete-time systems. There still remains a need for an efficient and high-performance belief space planning algorithm that is capable of directly handling systems with inherently continuous-space, continuous-time dynamics, such as maneuvering micro-aerial vehicles, or autonomous cars at freeways speeds.

1.2 Contributions

Our approach presented in this paper is significantly different than the previous approaches discussed above. We view the stochastic belief dynamics as a hybrid system with time-driven switching [18], where the controls are applied in continuous time and the observations are made in discrete time. A discrete-time observation creates a jump discontinuity in the belief state trajectory due to a sudden Bayesian update of the belief state. This view of belief space planning yields a continuous-time optimal control problem of a high-dimensional hybrid system. We then propose a model-based control algorithm to efficiently compute the control signals in a receding-horizon fashion. The algorithm is based on Sequential Action Control (SAC) [1]. SAC in its original form is a deterministic, model-based hybrid control algorithm, which "perturbs" a nominal control trajectory in a structured way so that the cost functional is optimally reduced up to the first order. The key to this approach is the use of perturbation theory of differential equations that is often discussed in the mode scheduling literature [19, 20]. As a result, SAC derives the optimal perturbation in closed form and synthesizes control signals at a high frequency to achieve a significant improvement over other optimal control methods based on local trajectory optimization.

We apply the perturbation theory to parametric Bayesian filters and derive the optimal control perturbation using the framework of SAC. To account for stochasticity, we also extend the original algorithm by incorporating Monte Carlo sampling of nominal belief trajectories. Our key contribution is the resulting continuous belief space planning algorithm, which we name SACBP. The algorithm has the following desirable properties:

1. SACBP optimizes the expected value of the first-order reduction of the cost functional with respect to some nominal control in near real-time.
2. SACBP does not require discretization of the state space, the observation space, or the control space. It also does not require discretization of time other than for numerical integration purposes.
3. General nonlinear parametric Bayesian filters can be used for state estimation as long as the system is control-affine and the control cost is quadratic.
4. Stochasticity in the future observations are fully considered.
5. SACBP is an anytime algorithm. Furthermore, the Monte Carlo sampling part of the algorithm is naturally parallelizable.
6. Even though SACBP is inherently suboptimal for the original stochastic optimal control problem, empirical results suggest that it is highly sample-efficient and outperforms other approaches when near real-time performance is required.

Although there exists prior work [21] that uses SAC for active sensing, its problem formulation relies on the ergodic control framework, which is significantly different from the belief space planning framework we propose here. We show that our SACBP outperforms ergodic trajectory optimization, MCTS-DPW, and a greedy method on a multi-target tracking example. We also show that SACBP outperforms belief iLQG and MCTS-DPW on a manipulation scenario. In the next section we derive relevant equations and present the SACBP algorithm along with a running time analysis. The simulation results are summarized in Section 3, followed by conclusions and future work in Section 4.

2 SACBP Algorithm

We first consider the case where some components of the state are fully observable. This mixed observability assumption is common in various active sensing problems [7, 22, 23] where the state of the robot is perfectly known, but some external variable of interest (e.g. a target's location) is stochastic. In addition, deterministic state transitions are often assumed for the robot. Therefore, in Section 2.1 we derive the SACBP control update formulae for this case. The general belief space planning where none of the state is fully observable or deterministically controlled is discussed in Section 2.2. An extension to use a closed-loop policy as the nominal control is presented in Section 2.3. The computational time complexity is discussed in Section 2.4.

2.1 Problems with Mixed Observability

Suppose that a robot can fully observe and deterministically control some state $p(t) \in \mathbb{R}^n$. Other states are not known to the robot and are estimated with the belief vector $b(t) \in \mathbb{R}^m$. We define the augmented state as $s \triangleq (p^{\mathrm{T}}, b^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{n+m}$.

Dynamics Model The physical state p is described by the following ODE:

$$\dot{p}(t) = f\left(p(t), u(t)\right), \tag{1}$$

where $u(t) \in \mathbb{R}^l$ is the control signal. On the other hand, suppose that the belief state changes in discrete time upon arrival of a new observation from the sensors. We will discuss the more general continuous-discrete filtering in Section 2.2. Let t_k be the time when k-th observation becomes available to the robot. The belief state transition is given by

$$\begin{cases} b(t_k^+) = g(p(t_k^-), b(t_k^-), y_k) \\ b(t) = b(t_k^+) & t \in [t_k^+, t_{k+1}^-], \end{cases} \tag{2}$$

where t_k^+ is infinitesimally larger than t_k and t_k^- smaller. Nonlinear function g corresponds to a parametric Bayesian filter (e.g., Kalman Filter, Extended Kalman Filter, Discrete Bayesian Filter, etc.) that takes the new observation $y_k \in \mathbb{R}^q$ and returns the updated belief state. The concrete choice of the filter depends on the instance of the problem. We demand the Bayesian filter to be differentiable in p and b, which we assume hereafter.[1]

Equations (1) and (2) constitute a hybrid system with time-driven switching [18]. This hybrid system representation is practical since it captures the fact that the observation updates occur less frequently than the control actuation in general, due to expensive information processing of sensor readings. Furthermore, with this representation one can naturally handle agile systems as they are without coarse discretization in time.

Given the initial state $s(t_0) = (p(t_0)^{\mathrm{T}}, b(t_0)^{\mathrm{T}})^{\mathrm{T}}$ and a control trajectory from t_0 to t_f denoted as $u(t_0 \to t_f)$, the system evolves stochastically according to the hybrid dynamics equations. The stochasticity is due to a sequence of stochastic future observations that will be taken by t_f.[2]

Perturbed Dynamics The control synthesis of SACBP begins with a given nominal control trajectory $u_n(t_0 \to t_f)$. Suppose that the nominal control is applied to the system and a sequence of K observations $(y_{(1)}, \dots, y_{(K)})$ is obtained. Conditioned on the observation sequence, the augmented state evolves deterministically. Let $s_n = (p_n^{\mathrm{T}}, b_n^{\mathrm{T}})^{\mathrm{T}}$ be the nominal trajectory of the augmented state induced by $(y_{(1)}, \dots, y_{(K)})$.

Now let us consider perturbing the nominal trajectory at a fixed time τ for a short duration $\epsilon > 0$. The perturbed control trajectory u_w is defined as

$$u_w(t) \triangleq \begin{cases} w & \text{if } t \in [\tau - \epsilon, \tau] \\ u_n(t) & \text{otherwise.} \end{cases} \tag{3}$$

[1] Prior work such as [4, 10] also assumes this differentiability property.

[2] We assume that the observation interval $t_{k+1} - t_k \triangleq \Delta t_o$ is fixed, and the control signals are recomputed when a new observation is incorporated in the belief.

The resulting perturbed system trajectory can be written as

$$\begin{cases} p_w(t,\epsilon) \triangleq p_n(t) + \epsilon \Psi_p(t) + o(\epsilon) \\ b_w(t,\epsilon) \triangleq b_n(t) + \epsilon \Psi_b(t) + o(\epsilon), \end{cases} \tag{4}$$

where $\Psi_p(t) \triangleq \frac{\partial_+}{\partial \epsilon} p_w(t,\epsilon)\big|_{\epsilon=0}$ and $\Psi_b(t) \triangleq \frac{\partial_+}{\partial \epsilon} b_w(t,\epsilon)\big|_{\epsilon=0}$ are the state variations that are linear in the perturbation duration ϵ. The notation $\frac{\partial_+}{\partial \epsilon}$ represents the right derivative with respect to ϵ. The state variations at perturbation time τ satisfy

$$\begin{cases} \Psi_p(\tau) = f(p_n(\tau), w) - f(p_n(\tau), u_n(\tau)) \\ \Psi_b(\tau) = 0, \end{cases} \tag{5}$$

assuming that τ does not exactly correspond to one of the switching times t_k [24]. For $t \geq \tau$, the physical state variation Ψ_p evolves according to the following first-order ODE:

$$\dot{\Psi}_p(t) = \frac{d}{dt}\left(\frac{\partial_+}{\partial \epsilon} p_w(t,\epsilon) \bigg|_{\epsilon=0} \right) \tag{6}$$

$$= \frac{\partial_+}{\partial \epsilon} f(p_w(t,\epsilon), u_n(t)) \bigg|_{\epsilon=0} \tag{7}$$

$$= \frac{\partial}{\partial p} f\left(p_n(t), u_n(t)\right) \Psi_p(t), \tag{8}$$

where the chain rule of differentiation and $p_w(t,0) = p_n(t)$ are used in (8). The dynamics of the belief state variation Ψ_b in the continuous region $t \in [t_k^+, t_{k+1}^-]$ are $\dot{\Psi}_b(t) = 0$ since the belief vector $b(t)$ is constant according to (2). However, across the discrete jumps the belief state variation Ψ_b changes discontinuously and satisfies

$$\Psi_b(t_k^+) = \frac{\partial_+}{\partial \epsilon} b_w(t_k^+, \epsilon) \bigg|_{\epsilon=0} \tag{9}$$

$$= \frac{\partial_+}{\partial \epsilon} g\left(p_w(t_k^-, \epsilon), b_w(t_k^-, \epsilon), y_k\right) \bigg|_{\epsilon=0} \tag{10}$$

$$= \frac{\partial}{\partial p} g\left(p_n(t_k^-), b_n(t_k^-), y_k\right) \Psi_p(t_k^-) + \frac{\partial}{\partial b} g\left(p_n(t_k^-), b_n(t_k^-), y_k\right) \Psi_b(t_k^-). \tag{11}$$

Perturbed Cost Functional Let us consider the cost functional of the form

$$J(p,b,u) = \int_{t_0}^{t_f} c\left(p(t), b(t), u(t)\right) dt + h(p(t_f), b(t_f)), \tag{12}$$

where c is the running cost and h is the terminal cost. Following the discussion above on the perturbed dynamics, let J_n be the total cost of the nominal trajectory conditioned on the given observation sequence $(y_{(1)}, \ldots, y_{(K)})$. Under the fixed perturbation time τ, we represent the perturbed cost J_w in terms of J_n as

$$J_w(p_w, b_w, \epsilon) \triangleq J_n + \epsilon\nu(t_f) + o(\epsilon), \tag{13}$$

where $\nu(t_f) \triangleq \frac{\partial_+}{\partial\epsilon} J_w(p_w, b_w, \epsilon)|_{\epsilon=0}$ is the variation of the cost functional linear in ϵ. For further analysis it is convenient to express the running cost in the Mayer form [24]. Let $s^0(t)$ be a new state variable defined by $\dot{s}^0(t) = c\left(p(t), b(t), u(t)\right)$ and $s(t_0) = 0$. Then the total cost is a function of the appended augmented state $\bar{s} \triangleq (s^0, s^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{1+n+m}$ at time t_f, which is given by

$$J = s^0(t_f) + h\left(s(t_f)\right). \tag{14}$$

Using this form of total cost J, the perturbed cost (13) is

$$J_w = J_n + \epsilon \begin{bmatrix} 1 \\ \frac{\partial}{\partial p} h\left(p_n(t_f), b_n(t_f)\right) \\ \frac{\partial}{\partial b} h\left(p_n(t_f), b_n(t_f)\right) \end{bmatrix}^{\mathrm{T}} \bar{\Psi}(t_f) + o(\epsilon), \tag{15}$$

where $\bar{\Psi}(t_f) \triangleq \left(\Psi^0(t_f)^{\mathrm{T}}, \Psi_p(t_f)^{\mathrm{T}}, \Psi_b(t_f)^{\mathrm{T}}\right)^{\mathrm{T}}$. Note that the dot product in (15) corresponds to $\nu(t_f)$ in (13). The variation of the appended augmented state Ψ^0 follows the variational equation

$$\dot{\Psi}^0(t) = \frac{d}{dt}\left(\left.\frac{\partial_+}{\partial\epsilon} s_w^0(t, \epsilon)\right|_{\epsilon=0}\right) \tag{16}$$

$$= \frac{\partial}{\partial p} c(p_n(t), b_n(t), u_n(t))^{\mathrm{T}} \Psi_p(t) + \frac{\partial}{\partial b} c(p_n(t), b_n(t), u_n(t))^{\mathrm{T}} \Psi_b(t). \tag{17}$$

The perturbed cost equation (15), especially the dot product expressing $\nu(t_f)$, is consequential; it tells us how the total cost functional changes due to the perturbation at some fixed time τ, up to the first order with respect to the perturbation duration ϵ. At this point, one could compute the value of $\nu(t_f)$ for a control perturbation with a specific value of (w, τ) by simulating the nominal dynamics and integrating the variational equations (8)(11)(17) up to t_f.

Adjoint Equations Unfortunately, this forward integration of $\nu(t_f)$ is not so useful by itself since we are interested in finding the value of (w, τ) that leads to the minimum value of $\nu(t_f)$, if it exists; it would be computationally intensive to apply control perturbation at different application times τ with different values of w and re-simulate state variation $\bar{\Psi}$. To avoid this computationally expensive search, Ansari and Murphey [1] has introduced the adjoint system $\bar{\rho}$ with which the dot product is invariant: $\frac{d}{dt}\left(\bar{\rho}(t)^{\mathrm{T}}\bar{\Psi}(t)\right) = 0 \ \forall t \in [t_0, t_f]$. If we let $\bar{\rho}(t_f) = \left(1, \frac{\partial}{\partial p} h\left(p_n(t_f), b_n(t_f)\right)^{\mathrm{T}}, \frac{\partial}{\partial b} h\left(p_n(t_f), b_n(t_f)\right)^{\mathrm{T}}\right)^{\mathrm{T}}$ so that its dot product with $\bar{\Psi}(t_f)$ equals $\nu(t_f)$ as in (15), the time invariance gives

$$\nu(t_f) = \bar{\rho}(\tau)^{\mathrm{T}} \bar{\Psi}(\tau) \tag{18}$$

$$= \bar{\rho}(\tau)^{\mathrm{T}} \begin{bmatrix} c\left(p_n(\tau), b_n(\tau), w\right) - c\left(p_n(\tau), b_n(\tau), u_n(\tau)\right) \\ f(p_n(\tau), w) - f(p_n(\tau), u_n(\tau)) \\ 0 \end{bmatrix}. \tag{19}$$

Therefore, we can compute the first-order cost change $\nu(t_f)$ for different values of τ once the adjoint trajectory is derived. For $t \in [t_k^+, t_{k+1}^-]$ the time derivative of Ψ_b exists, and the invariance property suggests that $\dot{\bar{\rho}}(t)^{\mathrm{T}}\bar{\Psi}(t) + \bar{\rho}(t)^{\mathrm{T}}\dot{\bar{\Psi}}(t) = 0$ is enforced. It can be verified that the following system satisfies this equation:

$$\begin{cases} \dot{\rho}^0(t) = 0 \\ \dot{\rho}_p(t) = -\frac{\partial}{\partial p} c(p_n(t), b_n(t), u_n(t)) - \frac{\partial}{\partial p} f(p_n(t), u_n(t))^{\mathrm{T}} \rho_p(\tau) \\ \dot{\rho}_b(t) = -\frac{\partial}{\partial b} c(p_n(t), b_n(t), u_n(t)). \end{cases} \tag{20}$$

Analogously, at discrete jumps we can still enforce the invariance by setting $\bar{\rho}(t_k^+)^{\mathrm{T}}\bar{\Psi}(t_k^+) = \bar{\rho}(t_k^-)^{\mathrm{T}}\bar{\Psi}(t_k^-)$, which holds for the following adjoint equations:

$$\begin{cases} \rho^0(t_k^-) = \rho^0(t_k^+) \\ \rho_p(t_k^-) = \rho_p(t_k^+) + \frac{\partial}{\partial p} g\left(p_n(t_k^-), b_n(t_k^-), y_k\right)^{\mathrm{T}} \rho_b(t_k^+) \\ \rho_b(t_k^-) = \frac{\partial}{\partial b} g\left(p_n(t_k^-), b_n(t_k^-), y_k\right)^{\mathrm{T}} \rho_b(t_k^+). \end{cases} \tag{21}$$

Note that the adjoint system integrates backward in time as it has the boundary condition defined at t_f. More importantly, the adjoint dynamics (20)(21) only depend on the nominal trajectory of the system (p_n, b_n) and the observation sequence $(y_{(1)}, \ldots, y_{(K)})$. The linear variation term $\nu(t_f)$ is finally given by

$$\begin{aligned} \nu(t_f) = c(p_n(\tau), b_n(\tau), w) - c(p_n(\tau), b_n(\tau), u_n(\tau)) + \\ \rho_p(\tau)^{\mathrm{T}} \left(f(p_n(\tau), w) - f(p_n(\tau), u_n(\tau))\right). \end{aligned} \tag{22}$$

Control Optimization In order to efficiently optimize (22) with respect to (w, τ), the rest of the paper assumes that the control cost is quadratic $\frac{1}{2}u^{\mathrm{T}}C_u u$ and the dynamics model $f(p, u)$ is control-affine with linear term $H(p)u$. Although the control-affine assumption may appear restrictive, many physical systems possess this property in engineering practice. As a result of these assumptions, (22) becomes

$$\nu(t_f) = \frac{1}{2}w^{\mathrm{T}}C_u w + \rho_p(\tau)^{\mathrm{T}} H(p_n(\tau))(w - u_n(\tau)) - \frac{1}{2}u_n(\tau)^{\mathrm{T}} C_u u_n(\tau). \tag{23}$$

So far we have treated the observation sequence $(y_{(1)}, \ldots, y_{(K)})$ as given and fixed. However, in practice it is a random process that we have to take into account. Fortunately, our control optimization is all based on the nominal control $u_n(t_0 \to t_f)$, with which we can both simulate the augmented dynamics and sample the observations. To see this, let us rewrite $\nu(t_f)$ in (23) as $\nu(t_f, y_{(1)}, \ldots, y_{(K)})$ to clarify the dependence on the observations. The expected value of the first order cost variation is given by

$$\mathbb{E}[\nu(t_f)] = \int \nu(t_f, y_{(1)}, \ldots, y_{(K)}) p\left(y_{(1)}, \ldots, y_{(K)} \mid u_n(t_0 \to t_f)\right) dy_{(1)} \ldots dy_{(K)}. \tag{24}$$

Even though we do not know the values of the distribution above, we have the generative model; we can simulate the augmented state trajectory using the nominal control $u_n(t_0 \rightarrow t_f)$ and sample the stochastic observations from the belief states along the trajectory.

Using the linearity of expectation for (23), we have

$$\mathbb{E}[\nu(t_f)] = \frac{1}{2}w^{\mathrm{T}}C_u w + \mathbb{E}[\rho_p(\tau)]^{\mathrm{T}}H(p_n(\tau))(w - u_n(\tau)) - \frac{1}{2}u_n(\tau)^{\mathrm{T}}C_u u_n(\tau). \tag{25}$$

Notice that only the adjoint trajectory is stochastic. We can employ Monte Carlo sampling to sample a sufficient number of observation sequences to approximate the expected adjoint trajectory. Now (25) becomes a convex quadratic in w for $C_u \succ 0$. An existing convex solver efficiently solves the following convex program with a box saturation constraint. Furthermore, analytical solutions are available if C_u is diagonal, since the coordinates of w are completely decoupled in this case.

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \mathbb{E}[\nu(t_f)] \\ \text{subject to} \quad & a \preceq w \preceq b \end{aligned} \tag{26}$$

This optimization is solved for different values of $\tau \in (t_0 + t_{calc} + \epsilon, t_0 + \Delta t_o)$, where t_{calc} is the pre-allocated computational time budget and Δt_o is the time interval between two successive observations and control updates. We then search for the optimal perturbation time τ^* to globally minimize $\mathbb{E}[\nu(t_f)]$ over $(w^*(\tau), \tau)$. There is only a finite number of τ to consider since in practice we use numerical integration such as the Euler scheme with some step size Δt_c to compute the trajectories. In [1] the finite perturbation duration ϵ is also optimized using line search, but in this work we set ϵ as a tunable parameter to reduce the computational complexity. The complete algorithm is summarized in Algorithm 1, which is called every Δt_o[s] as the new observation is incorporated in the belief.

2.2 General Belief Space Planning Problems

If none of the state is fully observable, the same stochastic SAC framework still applies almost as is to the belief sate b. In this case we consider a continuous-discrete filter [25] where the prediction step follows an ODE and the update step provides an instantaneous discrete jump. The hybrid dynamics for the belief vector are given by

$$\begin{cases} b(t_k^+) = g(b(t_k^-), y_k) \\ \dot{b}(t) = f_b(b(t), u(t)) \quad & t \in [t_k^+, t_{k+1}^-]. \end{cases} \tag{27}$$

Mirroring the approach in Section 2.1, one can verify that the linear cost variation term $\nu(t_f)$ has the same form as (22):

$$\nu(t_f) = c(b_n(\tau), w) - c(b_n(\tau), u_n(\tau)) + \rho(\tau)^{\mathrm{T}} \left(f_b(b_n(\tau), w) - f_b(b_n(\tau), u_n(\tau)) \right). \tag{28}$$

Algorithm 1 SACBP Control Update

```
INPUT: Current augmented state $s(t_0) = (p(t_0)^{\mathrm{T}}, b(t_0)^{\mathrm{T}})^{\mathrm{T}}$ or belief state $b(t_0)$,
    nominal control trajectory $u_n(t_0 \to t_f)$, perturbation duration $\epsilon > 0$
OUTPUT: Optimally perturbed control trajectory $u_w(t_0 \to t_f)$
 1: for $i = 1{:}N$ do
 2:     Forward-simulate augmented state trajectory (1)(2) or belief state trajectory (27) and sample observation sequence $(y^i_{(1)}, \dots, y^i_{(K)})$ from the belief states.
 3:     Backward-simulate adjoint trajectory $\rho^i(t_0 \to t_f)$ (20)(21) or (29) with sampled observations.
 4: end for
 5: Monte Carlo estimate $\mathbb{E}[\rho_p] \approx \frac{1}{N}\sum_{i=1}^{N} \rho^i_p$ or $\mathbb{E}[\rho^{\mathrm{T}} H_b(b_n)] \approx \frac{1}{N}\sum_{i=1}^{N} \rho^{i\mathrm{T}} H_b(b^i_n)$.
 6: for $(\tau = t_0 + t_{calc} + \epsilon;\ \tau \le t_0 + \Delta t_o;\ \tau \leftarrow \tau + \Delta t_c)$ do
 7:     Solve convex program (26). Store optimal value $v^*(\tau)$ and optimizer $w^*(\tau)$.
 8: end for
 9: $\tau^* \leftarrow \arg\min v^*(\tau),\ w^* \leftarrow w^*(\tau^*)$
10: $u_w(t_0 \to t_f) \leftarrow PerturbControlTrajectory(u_n, w^*, \tau^*, \epsilon)$ (3)
11: return $u_w(t_0 \to t_f)$
```

The adjoint variable ρ now has the same dimension as b and follows the adjoint dynamics

$$\begin{cases} \rho(t_k^-) = \frac{\partial}{\partial b} g(b_n(t_k^+), y_k)^{\mathrm{T}} \rho(t_k^+) \\ \dot{\rho}(t) = -\frac{\partial}{\partial b} c(b_n(t), u_n(t)) - \frac{\partial}{\partial b} f_b(b_n(t), u_n(t))^{\mathrm{T}} \rho(t), \end{cases} \tag{29}$$

with the boundary condition $\rho(t_f) = \frac{\partial}{\partial b} h(b_n(t_f))$. Under the control-affine assumption for f_b and the quadratic control cost,[3] the expected first order cost variation (28) yields

$$\mathbb{E}[\nu(t_f)] = \frac{1}{2} w^{\mathrm{T}} C_u w + \mathbb{E}[\rho(\tau)^{\mathrm{T}} H_b(b_n(\tau))](w - u_n(\tau)) - \frac{1}{2} u_n(\tau)^{\mathrm{T}} C_u u_n(\tau), \tag{30}$$

where H_b is the control coefficient term in f_b. We can sample $\rho(\tau)^{\mathrm{T}} H_b(b_n(\tau))$ via the forward-backward simulation of the dynamics.

2.3 Closed-loop Nominal Policy

In Sections 2.1 and 2.2 we assumed that the nominal control u_n was an open-loop control trajectory. However, one can think of a scenario where a nominal control

[3] Although it is difficult to state the general conditions under which this control-affine assumption holds, it can be verified that the continuous-discrete EKF [25] satisfies this property if the underlying state dynamics are control-affine.

is a closed-loop policy computed off-line, possibly using a POMDP algorithm in a discretized space. Indeed, SACBP can also handle closed-loop nominal policies. Let π_n be a closed-loop nominal policy, which is a mapping from either an augmented state s or a belief state b to a control value u. Due to the stochastic belief dynamics, the control values returned by π_n in the future is also stochastic. This is reflected when we take expectations over the nominal dynamics. Specifically, the terms dependent on u_n in (25) and (30) now become stochastic and thus need to be sampled. However, the equations are still convex quadratic in w as it is decoupled from the nominal control. Therefore, only lines 5 and 7 of Algorithm 1 are affected.

2.4 Computational Time Analysis

Let us analyze the time complexity of the SACBP algorithm. The bottleneck of the computation is when the forward-backward simulation is performed multiple times (lines 1–5 of Algorithm 1). The asymptotic complexity of this part is given by $O(N(\frac{t_f - t_0}{\Delta t_o})(M_{\text{forward}} + M_{\text{backward}}))$, where M_{forward} and M_{backward} are the times to respectively integrate the forward and backward dynamics between two successive observations. For a more concrete analysis let us use the Gaussian belief dynamics given by EKF as an example. For simplicity we assume the same dimension n for the state, control and the observation. The belief state has dimension $O(n^2)$. Using the Euler scheme, The forward integration takes $M_{\text{forward}} = O((\frac{\Delta t_o}{\Delta t_c} + 1)n^3)$ since evaluating continuous and discrete EKF equations are both $O(n^3)$. Evaluating the continuous part of the costate dynamics (29) is dominated by the computation of Jacobian $\frac{\partial f_b}{\partial b}$, which is $O(n^5)$ because $O(n^3)$ operations to evaluate f_b are carried out $O(n^2)$ times. The discrete part is also $O(n^5)$. Therefore, $M_{\text{backward}} = O((\frac{\Delta t_o}{\Delta t_c} + 1)n^5)$. Overall, the time complexity is $O(N(\frac{t_f - t_0}{\Delta t_o})(\frac{\Delta t_o}{\Delta t_c} + 1)n^5)$. This is asymptotically smaller in n than belief iLQG, which is $O(n^6)$. See [11] for a comparison of time complexity among different belief space planning algorithms. We also remind the readers that SACBP is an online method and a naive implementation already achieves near real-time performance, computing control in less than 0.4[s]. By near real-time we mean that a naive implementation of SACBP requires approximately $3 \times t_{\text{calc}}$ to $7 \times t_{\text{calc}}$ time to compute an action that must be applied t_{calc} in the future. We expect that parallelization in a GPU and a more efficient implementation will result in real-time computation for SACBP.

3 Simulation Results

We evaluated the performance of SACBP in the following simulation studies: (i) active multi-target tracking with range-only observations; (ii) object manipulation under model uncertainty. All the computation was performed on a desktop computer with Intel Core i7-6800K CPU and 62.8GB RAM. The Monte Carlo sampling of SACBP was parallelized on the CPU.

3.1 Active Multi-Target Tracking with Range-only Observations

This problem focuses on pure information gathering, namely identifying where the moving targets are in the environment. In doing so, the surveillance robot modeled as a single integrator can only use relative distance observations. The robot's position p is fully observable and transitions deterministically. Assuming perfect data association, the observation for target i is $d_i = ||q_i - p + v_i||_2$, where q_i is the true target position and v_i is zero-mean Gaussian white noise with state-dependent covariance $R(p, q_i) = R_0 + ||q_i - p||_2 R_1$. We used $0.01 I_{2\times 2}$ for the nominal noise R_0. The range-dependent noise $R_1 = 0.001 I_{2\times 2}$ degrades the observation quality as the robot gets farther from the target. The discrete-time UKF was employed for state estimation in tracking 20 independent targets. The target dynamics are modeled by a 2D Brownian motion with covariance $Q = 0.1 I_{2\times 2}$. Similarly to [26], an approximated observation covariance $R(p, \mu_i)$ was used in the filter to obtain tractable estimation results, where μ_i is the most recent mean estimate of q_i.

SACBP algorithm generated the continuous robot trajectory over 200[s] with planning horizon $t_f - t_0 = 2$[s], update interval $\Delta t_o = 0.2[s]$, perturbation duration $\epsilon = 0.16[s]$, and $N = 10$ Monte Carlo samples. The Euler scheme was used for integration with $\Delta t_c = 0.01[s]$. The Jacobians and the gradients were computed either analytically or using an automatic differentiation tool [27] to retain both speed and precision. In this simulation $t_{\text{calc}} = 0.05[s]$ was assumed no matter how long the actual control update took. We used $c(p, b, u) = 0.05 u^{\mathrm{T}} u$ for the running cost and $h(p, b) = \sum_{i=1}^{20} \exp(\text{entropy}(b_i))$ for the terminal cost, with an intention to reduce the worst-case uncertainty among the targets. The nominal control was constantly 0.

We compared SACBP against three benchmarks: (i) a greedy algorithm based on the gradient descent of terminal cost h, similar to [7]; (ii) MCTS-DPW [13, 28] in the Gaussian belief space; (iii) projection-based trajectory optimization for ergodic exploration [29–31]. We also implemented the belief iLQG algorithm, but the policy did not converge for this problem. We speculate that the non-convex terminal cost h contributed to this behavior, which in fact violates one of the underlying assumptions made in the paper [10].

MCTS-DPW used the same planning horizon as SACBP, however it drew $N = 15$ samples from the belief tree so the computation time of the two algorithms match approximately. Ergodic trajectory optimization is not a belief space planning approach but has been used in active sensing literature. Beginning with nominal control 0, it locally optimized the ergodicity of the trajectory with respect to the spatial information distribution based on Fisher information. This optimization was open-loop since the future observations were not considered. As a new observation became available, the distribution and the trajectory were recomputed. All the controllers were saturated at the same limit. The results presented in Fig. 1 clearly indicates a significant performance improvement of SACBP while achieving near real-time computation. More notably, SACBP generated a trajectory that periodically revisited the two groups whereas other methods failed to do so (Fig. 2).

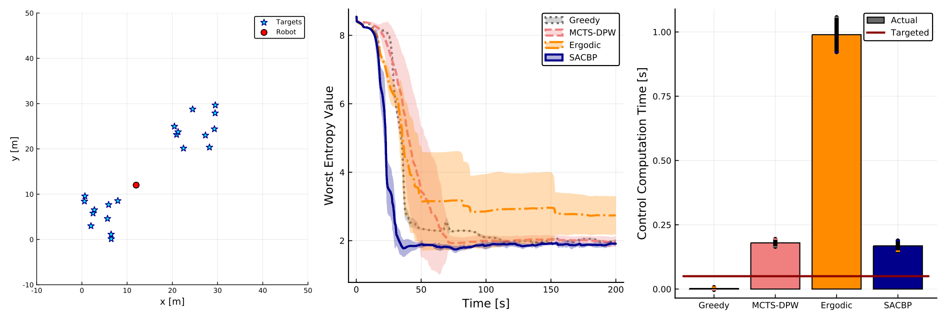

Fig. 1: (Left) Simulation environment with 20 targets and a surveillance robot. (Middle) The history of the worst entropy value among the targets averaged over 20 runs with the standard deviation. With the budget of 10 Monte Carlo samples, SACBP had small variance and consistently outperformed the other benchmarks on average. (Right) Computational time of SACBP achieved a reasonable value compared with the benchmarks, only 0.11[s] slower than the targeted value, i.e., simulated t_{calc}.

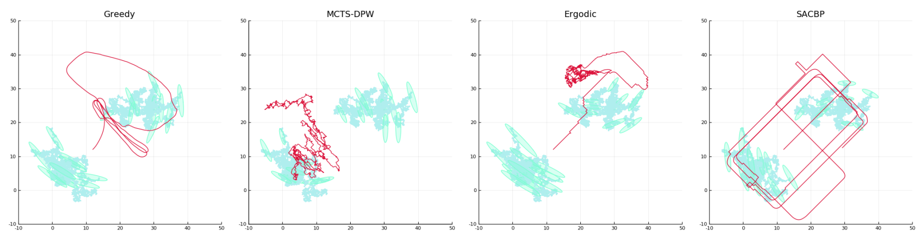

Fig. 2: Sample robot trajectories (depicted in red) generated by each algorithm. Greedy, MCTS-DPW, and Ergodic did not result in a trajectory that fully covers the two groups of the targets, whereas SACBP periodically revisited both of them. The blue lines are the target trajectories. The ellipses are 99% error ellipses.

3.2 Object Manipulation under Model Uncertainty

This problem is the model-based Bayesian reinforcement learning problem studied in [2], therefore the description of the nonlinear dynamics and the observation models are omitted. See Fig. 3 for the illustration of the environment. The robot applies the force and the torque to move the object whose mass, moment of inertia, moment arm lengths, and linear friction coefficient are unknown. The robot's state also needs to be estimated. The same values for $t_f - t_0$, $\varDelta t_o$, $\varDelta t_c$, t_{calc} as in the previous problem were assumed. SACBP used $\epsilon = 0.04[s]$ and $N = 10$. The nominal control was a closed-loop position controller whose input was the mean x-y position and the rotation estimates. The cost function was quadratic in the true state x and control u, given by $\frac{1}{2}x^{\mathrm{T}}C_x x + \frac{1}{2}u^{\mathrm{T}}C_u u$. Taking expectations yielded the equivalent cost in the Gaussian belief space

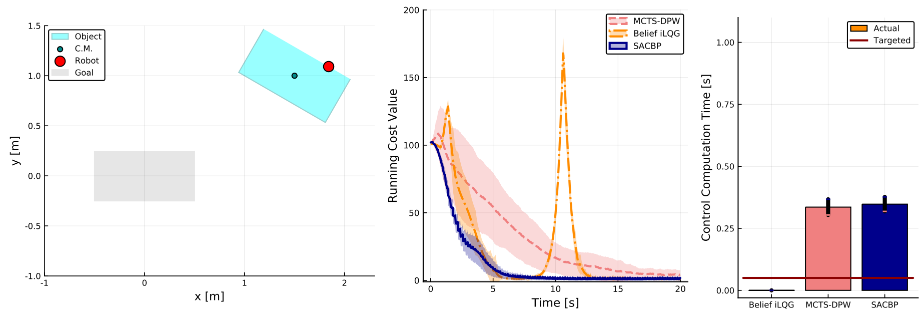

Fig. 3: (Left) The robot is attached to the rectangular object. (Middle) The history of the true running cost $\frac{1}{2}x^{\mathrm{T}}C_x x + \frac{1}{2}u^{\mathrm{T}}C_u u$ averaged over 20 cases. SACBP with $N = 10$ samples successfully brought the cost to almost 0, meaning that the goal was reached. MCTS-DPW with $N = 190$ was not as successful. Belief iLQG resulted in large overshoots around 2[s] and 11[s]. (Right) Computational time of SACBP increased from the multi-target tracking problem due to additional computation related to the closed-loop nominal policy. Note that belief iLQG took 5945[s] to derive the policy off-line, although the average online execution time was only 4.0×10^{-5}[s] per iteration.

$c(b, u) = \frac{1}{2}\mu^{\mathrm{T}}C_x\mu + \frac{1}{2}\mathrm{tr}(C_x\Sigma) + \frac{1}{2}u^{\mathrm{T}}C_u u$, where Σ is the covariance matrix. We let terminal cost h be the same as c without the control term.

We compared SACBP against (i) MCTS-DPW in the Gaussian belief space and (ii) belief iLQG. We allowed MCTS-DPW to draw $N = 190$ samples to set the computation time comparable to SACBP. As suggested in [2], MCTS-DPW used the position controller mentioned above as the rollout policy. Similarly, belief iLQG was initialized with a nominal trajectory generated by the same position controller. Note that both MCTS-DPW and belief iLQG computed controls for the discrete-time models whereas SACBP directly used the continuous-time model. However the simulation was all performed in continuous time, which could explain the large overshoot of the belief iLQG trajectory in Fig. 3. Overall, the results presented above demonstrate that SACBP succeeded in this task with only 10 Monte Carlo samples, reducing the running cost to almost 0 within 10[s]. Although the computation time increased from the previous problem due to the usage of a closed-loop nominal policy, it still achieved near real-time performance and much shorter than belief iLQG, which took 5945[s] until convergence in our implementation.

4 Conclusions and Future Work

In this paper we have presented SACBP, a novel belief space planning algorithm for continuous-time dynamical systems. We have viewed the stochastic belief dynamics as a hybrid system with time-driven switching and derived the optimal control perturbation based on the perturbation theory of differential equations. The resulting algorithm extends the framework of SAC to stochastic belief dynamics and is highly parallelizable to run in near real-time. Through the extensive simulation study we have confirmed that SACBP outperforms other algorithms including a greedy algorithm, a local trajectory optimization method, and an approximate dynamic programming approach. In future work we will consider a distributed multi-robot version of SACBP as well as problems where both discrete and continuous actions exist. We also plan to address questions regarding the theoretical guarantees of this algorithm and provide additional case studies with more complex belief distributions.

Acknowledgements

This work was supported by the Toyota Research Institute (TRI). This article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. This work was also supported in part by NSF grant CMMI-1562335, and a JASSO fellowship. The authors are grateful for this support.

References

1. Ansari, A.R., Murphey, T.D.: Sequential Action Control: Closed-Form Optimal Control for Nonlinear and Nonsmooth Systems. IEEE Transactions on Robotics 32(5), 1196–1214 (2016)
2. Slade, P., Culbertson, P., Sunberg, Z., Kochenderfer, M.: Simultaneous active parameter estimation and control using sampling-based bayesian reinforcement learning. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 804–810 (2017)
3. Mihaylova, L., Lefebvre, T., Bruyninckx, H., Gadeyne, K., De Schutter, J.: Active Sensing for Robotics - A Survey. In: Proceedings of 5th International Conference on Numerical Methods and Applications. pp. 316–324 (2002)
4. Platt, R., Tedrake, R., Kaelbling, L., Lozano-Perez, T.: Belief space planning assuming maximum likelihood observations. In: Robotics Science and Systems Conference (RSS) (2010)
5. Bourgault, F., Makarenko, A., Williams, S., Grocholsky, B., Durrant-Whyte, H.: Information based adaptive robotic exploration. In: IEEE/RSJ International Conference on Intelligent Robots and System. vol. 1, pp. 540–545. IEEE (2002)
6. Seekircher, A., Laue, T., Röfer, T.: Entropy-based active vision for a humanoid soccer robot. In: RoboCup 2010: Robot Soccer World Cup XIV, vol. 6556 LNCS, pp. 1–12. Springer Berlin Heidelberg (2011)
7. Schwager, M., Dames, P., Rus, D., Kumar, V.: A multi-robot control policy for information gathering in the presence of unknown hazards. In: Robotics Research : The 15th International Symposium ISRR. pp. 455–472. Springer International Publishing (2017)
8. Erez, T., Smart, W.D.: A scalable method for solving high-dimensional continuous pomdps using local approximation. In: Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence. pp. 160–167. UAI'10, AUAI Press, Arlington, Virginia, United States (2010)
9. Patil, S., Kahn, G., Laskey, M., Schulman, J., Goldberg, K., Abbeel, P.: Scaling up gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation. In: WAFR. Springer Tracts in Advanced Robotics, vol. 107, pp. 515–533. Springer (2014)
10. van den Berg, J., Patil, S., Alterovitz, R.: Motion planning under uncertainty using iterative local optimization in belief space. The International Journal of Robotics Research 31(11), 1263–1278 (2012)

11. Rafieisakhaei, M., Chakravorty, S., Kumar, P.R.: T-lqg: Closed-loop belief space planning via trajectory-optimized lqg. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 649–656 (2017)
12. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artif. Intell. 101(1-2), 99–134 (1998)
13. Couëtoux, A., Hoock, J.B., Sokolovska, N., Teytaud, O., Bonnard, N.: Continuous Upper Confidence Trees. In: 2011 International Conference on Learning and Intelligent Optimization. pp. 433–445. Springer, Berlin, Heidelberg (2011)
14. Browne, C.B., Powley, E., Whitehouse, D., Lucas, S.M., Cowling, P.I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., Colton, S.: A survey of monte carlo tree search methods. IEEE Transactions on Computational Intelligence and AI in Games 4(1), 1–43 (2012)
15. Nishimura, H., Schwager, M.: Active motion-based communication for robots with monocular vision. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 2948–2955 (2018)
16. Somani, A., Ye, N., Hsu, D., Lee, W.S.: Despot: Online pomdp planning with regularization. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2. pp. 1772–1780. NIPS'13, Curran Associates Inc., USA (2013)
17. Sunberg, Z., Kochenderfer, M.J.: POMCPOW: an online algorithm for pomdps with continuous state, action, and observation spaces. CoRR abs/1709.06196 (2017)
18. Heemels, W., Lehmann, D., Lunze, J., Schutter, B.D.: Introduction to hybrid systems. In: Lunze, J., Lamnabhi-Lagarrigue, F. (eds.) Handbook of Hybrid Systems Control – Theory, Tools, Applications, chap. 1, pp. 3–30. Cambridge University Press (2009)
19. Egerstedt, M., Wardi, Y., Axelsson, H.: Transition-time optimization for switched-mode dynamical systems. IEEE Transactions on Automatic Control 51(1), 110–115 (2006)
20. Wardi, Y., Egerstedt, M.: Algorithm for optimal mode scheduling in switched systems. In: 2012 American Control Conference (ACC). pp. 4546–4551 (2012)
21. Mavrommati, A., Tzorakoleftherakis, E., Abraham, I., Murphey, T.D.: Real-time area coverage and target localization using receding-horizon ergodic exploration. IEEE Transactions on Robotics pp. 62–80 (2018)
22. Le Ny, J., Pappas, G.J.: On trajectory optimization for active sensing in Gaussian process models. In: Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference. pp. 6286–6292. IEEE (2009)
23. Popovi, M., Hitz, G., Nieto, J., Sa, I., Siegwart, R., Galceran, E.: Online informative path planning for active classification using uavs. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 5753–5758 (2017)
24. Liberzon, D.: Calculus of variations and optimal control theory: A concise introduction. Princeton University Press (2011)
25. Xie, L., Popa, D., Lewis, F.L.: Optimal and robust estimation: with an introduction to stochastic control theory. CRC press (2007)
26. Spinello, D., Stilwell, D.J.: Nonlinear estimation with state-dependent gaussian observation noise. IEEE Transactions on Automatic Control 55(6), 1358–1366 (2010)
27. Revels, J., Lubin, M., Papamarkou, T.: Forward-mode automatic differentiation in julia. arXiv:1607.07892 [cs.MS] (2016), https://arxiv.org/abs/1607.07892
28. Egorov, M., Sunberg, Z.N., Balaban, E., Wheeler, T.A., Gupta, J.K., Kochenderfer, M.J.: Pomdps. jl: A framework for sequential decision making under uncertainty. Journal of Machine Learning Research 18(26), 1–5 (2017)
29. Miller, L.M., Murphey, T.D.: Trajectory optimization for continuous ergodic exploration. In: American Control Conference (ACC), 2013. pp. 4196–4201. IEEE (2013)
30. Miller, L.M., Silverman, Y., MacIver, M.A., Murphey, T.D.: Ergodic exploration of distributed information. IEEE Transactions on Robotics 32(1), 36–52 (2016)
31. Dressel, L., Kochenderfer, M.J.: Tutorial on the generation of ergodic trajectories with projection-based gradient descent. IET Cyber-Physical Systems: Theory & Applications (2018)