Dynamic Coverage for Active Learning From Equilibrium

Ian Abraham, Ahalya Prabhakar and Todd D. Murphey

(Invited Paper)

***Abstract*—This paper develops KL-Ergodic Exploration from Equilibrium (KL-E^3), a method for robotic systems to integrate stability into actively generating informative measurements through ergodic exploration. Ergodic exploration enables robotic systems to indirectly sample from informative spatial distributions globally, avoiding local optima, and without the need to evaluate the derivatives of the distribution against the robot dynamics. Using hybrid systems theory, we derive a controller that allows a robot to exploit equilibrium policies (i.e., policies that solve a task) while allowing the robot to explore and generate informative data using an ergodic measure that can extend to high-dimensional states. We show that our method is able to maintain Lyapunov attractiveness with respect to the equilibrium task while actively generating data for learning tasks such, as Bayesian optimization, model learning, and off-policy reinforcement learning. In each example, we show that our proposed method is capable of generating an informative distribution of data while synthesizing smooth control signals. We illustrate these examples using simulated systems and provide simplification of our method for real-time online learning in robotic systems.**

Note to Practitioners: ***Abstract*—Robots need to learn and explore to collect measurement data in the real world without failing catastrophically. They need to adapt to collected measurements and exploit an understanding of physics such that we can enable them to truly begin their own experimentation in the real world. Standard learning methods do not have any restrictions on how the robot can explore and learn, making the robot dynamically volatile. Those that do, are often too restrictive in terms of the stability of the robot, resulting in a lack of improved learning due to poor data collection. Applying our method would allow robotic systems to be able to adapt online without the need for human intervention. We show that by taking into account both the dynamics of the robot and the statistics of where the robot has been, we are able to naturally encode where the robot needs to explore and collect measurements for efficient learning that is dynamically safe. With our method we are consistently able to effectively learn while being energetically efficient compared to state-of-the-art active learning methods. Our approach is able to do so in a single execution of the robotic system, i.e., the robot does not need human intervention to reset it. Further research works towards making the learned dynamic models of the robotic system more physically accurate. Future work will consider multi-agent robotic systems that actively learn and explore in a team of collaborative robots.**

***Index Terms*—Active Learning, Active Exploration, Online Learning, Stable Learning**

I. Introduction

ROBOT learning has proven to be a challenge for real-world application. This is partially due to the ineffectiveness of passive data acquisition for learning and the necessity for data-driven actions for collecting informative data. What makes this problem even more difficult is that generating data for robotic systems is often an unstable process. It involves generating measurements dependent upon physical motion of the robot. As a result, safe data collection through exploration becomes a challenge. The problem becomes exacerbated when memory and constraints (i.e., data must come from a single roll out) are imposed on the robot. Thus, robotic systems need to adapt subject to data in a systematic and informative manner while preserving a notion of equilibrium as a means of safety for itself, the environment, and potentially humans. In this paper, we address these issues by developing an algorithm that draws on hybrid systems theory [1] and ergodic exploration [2]–[7]. Our approach enables robots to generate and collect informative data while guaranteeing Lyapunov attractiveness [8] with respect to an equilibrium task.[1]

Actively collecting data and learning are often characterized as being part of the same problem of learning from experience [9], [10]. This is generally seen in the field of reinforcement learning (RL) where attempts at a task, as well as learning from the outcome of actions, are used to learn policies and predictive models [9], [11]. Much of the work in the field is dedicated towards generating a sufficiently large distribution of data such that it encompasses unforeseen events, allowing generalization to real-world application [9], [11]–[13] however inefficient the method. In this work, rather than trying to generate a large distribution of data given many attempts at a task, we seek to generate an informative data distribution, and learn from the collected distribution of data, as two separate problems, where we focus on actively and intelligently collecting data in a manner that is safe and efficient while adopting existing methods that enable learning.

Current safe learning methods typically provide some bound on the worst outcome model using probabilistic approaches [14], but often only consider the safety with respect to the task and not with respect to the exploration process. We focus on problems where exploring for data intersects with exploring the physical space of robots such that actions that are capable of generating the best set of data can destabilize the robot.

Ian Abraham, Ahalya Prabhakar, and Todd D. Murphey are all with the Department of Mechanical Engineering at Northwestern University, Evanston, IL, 60208. corresponding e-mail: (see i-abr@u.northwestern.edu

[1]Often such equilibrium tasks can be thought of stabilization, but can be viewed as running or executing a learned skill consistently like swinging up and stabilizing a cart pole.

In treating active exploration for data as an ergodic exploration problem, where the time spent during the trajectory of the robot is proportional to the measure of informative data in that regions, we are able to more efficiently use the physical motion of the robot by focusing on dynamic area coverage in the search space as opposed to directly generating samples from the most informative regions. With this approach, we are able to integrate known equilibria and feedback policies (we refer to these as equilibrium policies) which provide attractiveness guarantees—that the robot will eventually return to an equilibrium —while providing the control authority that allows the robot to actively seek out and collect informative data in order to later solve a learning task. Our contributions are summarized as follows:

- Developing a method which extends ergodic exploration to higher dimensional state-spaces using a sample-based measure.
- Synthesis of a control signal that exploits known equilibrium policies.
- Present theoretical results on the Lyapunov attractiveness centered around equilibrium policies of our method.
- Illustrate our method for improving the sample efficiency and quality of data collected for example learning goals.

We structure the paper as follows: Section II provides a list of related work, Section III defines the problem statement for this work. Section IV introduces our approximation to an ergodic metric and Section V formulates the ergodic exploration algorithm for active learning and exploration from equilibrium. Section VI provides examples where our method is applicable. Last, Section VII provides concluding remarks on our method and future directions.

II. Related Work

Active Exploration: Existing work generally formulates problems of active exploration as information maximization with respect to a known parameterized model [15], [16]. The problem with this approach is the abundance of local optima [2], [16] which the robotic system needs to overcome, resulting in insufficient data collection. Other approaches have sought to solve this problem by viewing information maximization as an area coverage problem [2], [4]. Ergodic exploration, in particular, has remedied the issue of local optima by using the ergodic metric to minimize the Sobelov distance [17] from the time-averaged statistics of the robot's trajectory to the expected information in the explored region. This enables both exploration (quickly in low information regions) and exploitation (spending significant amount of time in highly informative regions) in order to avoid local optima and harvest informative measurements. Our work utilizes this concept of ergodicity to improve how robotic systems explore and learn from data.

Ergodicity and Ergodic Exploration: A downside with the current methods for generating ergodic exploration in robots is that they assumes that the model of the robot is fully known. Moreover, there is little guarantee that the robot will not destabilize during the exploration process. This becomes an issue when the robot must explore part of its own state-space (i.e., velocity space) in order to generate informative data. Another issue is that these methods do not scale well with the dimensionality of the search space, making experimental applications with this approach challenging due to computational limitations. Our approach overcomes these issues by using a sample-based KL-divergence measure [4] as a replacement for the ergodic metric. This form of measure has been used previously; however, it relied on motion primitives in order to compute control actions [4]. We show that we can generate a continuous control signal that minimizes this ergodic measure using hybrid systems theory. The same approach is then shown to be readily amenable to existing equilibrium policies. As a result, we can use approximate models of dynamical systems instead of complete dynamic reconstructions in order to actively generate data while ensuring safety in the exploration process through a notion of attractiveness.

Off-Policy Learning: In this work, we utilize concepts from off-policy learning methods [18], [19] which are a set of methods that divides the learning and data generation phases. With these methods, data is first generated using some random policy and a value function is learned from rewards gathered. The value function is then used to create a policy which then updates the value function [20]. Generating more data does not require directly using the policy; however, the most common practice is to use the learned policy with added noise to guide the policy learning. These methods often rely on samples from a buffer of prior data during the training process rather than learning directly through the application of the policy. As such, they are more sample-efficient and can reuse existing data. A disadvantage with off-policy methods is that they are highly dependent on the distribution of data, resulting in an often unstable learning process. Our approach focuses on improving the data generation process through ergodic exploration to generate an informed distribution of data. As a result, the learning process is able to attain improved results from the generated distribution of data and retain its sample-efficiency.

Bayesian Optimization: Our work is most related to the structure of Bayesian optimization [21]–[23]. In Bayesian optimization, the goal is to find the maximum of an objective function which is unknown. At each iteration of Bayesian optimization, the unknown objective is sampled and a probabilistic model is generated. An acquisition function is then used as a metric for an "active learner" to find the next best sample. This loop repeats until a maximum is found. In our work, the "active learner" is the robot itself which must abide by the physics that governs its motion. As a result, the assumption that the active learner has the ability to sample anywhere in the search space is lost. Another difference is that instead of using a sample-based method to find the subsequent sampling position, as done in Bayesian optimization, we use ergodic exploration to generate a set of samples proportional to some spatial distribution. We pose the spatial distribution as an acquisition function which we show in Section VI-A. Thus, the active learner is able to sample from regions which have lower probability densities quickly and spending time in regions which are likely to produce an optima. Note that it is possible for one to directly use the derivatives of the acquisition function and a model of the robot's dynamics to

search for the best actions that a robot can take to sample from the objective; however, it is likely that similar issues with information maximization and local optima will occur.

Information Maximization: Last, we review work that addresses the data-inefficiency problem through information maximization [24]. These methods work by either direct maximization of an information measure or by pruning a data-set based on some information measure [25]. These methods still suffer from the problem of local minima due to a lack of exploration or non-convex information objectives [26]. Our work uses ergodicity as a way to gauge how much a robot should be sampling from the exploration space. As a result, the motion of the robot by minimizing an ergodic measure will automatically optimize where and for how long the robot should sample from, avoiding the need to prune from a data-set and sample from multiple highly informative peaks.

The following section introduces ergodicity and ergodic exploration and defines the problem statement for this work.

III. Ergodicity and The Ergodic Metric for Exploration and Exploitation

In this section, we first motivate ergodicity as an approach to the exploration vs. exploitation problem and the need for an ergodic measure. Then, we introduce ergodicity and ergodic exploration as the resulting outcome of optimizing an ergodic metric.

The exploration vs. exploitation problem is a problem in robot learning where the robot must deal with the choosing to exploit what it already knows or explore for more information, which entails running the risk of damaging itself or collecting bad data. Ergodic exploration treats the problem of exploration and exploitation as a problem of matching a spatial distributions to a time-averaged distribution—that is, the probability of a positive outcome given a state is directly related to the time spent at that state. Thus, more time is spent in regions where there are positive outcomes (exploitation) and quickly explores states where there is low probability of a positive outcome (exploration).

Definition 1. *Ergodicity, in robotics, is defined when a robot whose time-averaged statistics over its states is equivalent to the spatial statistics of an arbitrarily defined distribution that intersects those states.*

The exact specifications of a spatial statistic varies depending on the underlying task and is defined for different tasks in Sections VI. For now, let us define the time-averaged statistics of a robot by considering its trajectory $x(t) : \mathbb{R} \to \mathbb{R}^n$ $\forall t \in [t_0, t_f]$ generated through an arbitrary control process $u(t) : \mathbb{R} \to \mathbb{R}^m$.

Definition 2. *Given a search domain $\mathcal{S}^v \subset \mathbb{R}^{n+m}$ where $v \leq n+m$, the time-averaged statistics (i.e., the time the robot spends in regions of the search domain $\mathcal{S}^v$) of a trajectory $x(t)$ is defined as*

$$c(s \mid x(t)) = \frac{1}{t_f - t_0} \int_{t_0}^{t_f} \delta\left[s - \bar{x}(t)\right] dt, \tag{1}$$

where $s \in \mathcal{S}^v$ is a point in the search domain, $\bar{x}(t)$ is the component of the robot's state $x(t)$ that intersects the search domain, and $\delta[\cdot]$ is the Dirac delta function.

In general, the spatial statistics are defined through its probability density function $p(s)$ where $p(s) > 0$, and $\int_{\mathcal{S}^v} p(s)ds = 1$. Given a spatial distribution $p(s)$, we can calculate an ergodic metric as the distance between the Fourier decomposition of $p(s)$ and $c(s \mid x(t))$: [2]

$$\mathcal{E}(x(t)) = \sum_{k \in \mathbb{N}^v} \Lambda_k \left(c_k - p_k\right)^2 \tag{2}$$

where Λ_k is a weight on the harmonics defined in [27],

$$c_k = \frac{1}{t_f - t_0} \int_{t_0}^{t_f} F_k(x(t))dt,$$
$$p_k = \int_{\mathcal{S}^v} p(s)F_k(s)ds,$$

and $F_k(s)$ is the k^{th} Fourier basis function.

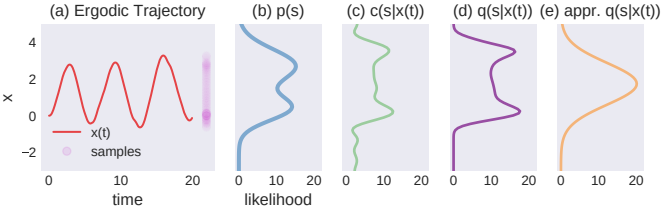


Fig. 1: (a) Illustration of an ergodic trajectory $x(t)$ with respect to (b) target distribution $p(s)$. Time-averaged distribution reconstructions of $x(t)$ are shown using (c) Definitions 2, (d) 3, and (e)(31). The Fourier decomposition approach often has residual artifacts due to the cosine approximation. 20 basis functions are used to approximate the time-averaged distribution. Σ-approximation to the time-averaged statistics minimizes residual artifacts. (e) Moment matching of the Σ-approximation provides a simplification for computing the time-averaged statistics.

As in [2], [28], one can calculate a controller that optimizes the ergodic metric such that the trajectory of the robot is ergodic with respect to the distribution $p(s)$ (see Fig 1 for illustration). However, this approach scales $\mathcal{O}(|k|^n)$ where $|k|$ is the maximum integer–valued Fourier term. As a result, this method is ill-suited for high-dimensional learning tasks whose exploration states are often the full state-space of the robot (often $n > 3$ for most mobile robots). Furthermore, the resulting time-averaged distribution reconstruction will often have residual artifacts which require additional conditioning to remove. This motivates the following section which defines an ergodic measure. [3]

IV. KL-Divergence Ergodic Measure

As an alternative to computing the ergodic metric, we present an ergodic measure which circumvents the scalability

[2]This distance is known as the Sobelov distance.

[3]We make note of the use of the work "measure" as opposed to metric as we allude to using the KL-divergence which itself is not a metric, but a measure.

issues mentioned in the previous section. To do this, we utilize the Kullback-Leibler divergence [4], [29], [30] (KL–divergence) as a measure for ergodicity. Let us first define the approximation to the time-averaged statistics of a trajectory $x(t)$:

Definition 3. *Given a search domain $\mathcal{S}^v \subset \mathbb{R}^{n+m}$ the Σ-approximated time-averaged statistics of the robot is defined by*

$$q(s \mid x(t)) = \frac{1}{t_f - t_0} \int_{t_0}^{t_f} \frac{1}{\eta} \exp\left[-\frac{1}{2}\|s - \bar{x}(t)\|^2_{\Sigma^{-1}}\right] dt \tag{3}$$

where $\Sigma \in \mathbb{R}^{v \times v}$ is a positive definite matrix parameter that specifies the width of the Gaussian and η is a normalization constant.

We call this an approximation because the true time-averaged statistics, as described in [2] and Definition 2, is a collection of delta functions parameterized by time. We approximate the delta function as a Gaussian distribution with variance Σ, converging as $\|\Sigma\| \to 0$. As an aside, one can treat Σ as a function of $\bar{x}(t)$ if there is uncertainty in the position of the robot.

With this approximation, we are able to relax the ergodic objective in [2] and use the following KL-divergence objective [4]:

$$\begin{aligned} D_{\mathrm{KL}}(p\|q) &= \int_{\mathcal{S}^v} p(s) \log \frac{p(s)}{q(s)} ds \\ &= \int_{\mathcal{S}^v} p(s) \log p(s) ds - \int_{\mathcal{S}^v} p(s) \log q(s) ds, \\ &= -\int_{\mathcal{S}^v} p(s) \log q(s) ds \\ &= -\mathbb{E}_{p(s)}\left[\log q(s)\right] \end{aligned}$$

where $\mathbb{E}$ is the expectation operator, $q(s) = q(s \mid x(t))$, and $p(s)$ is an arbitrary spatial distribution. Note that we drop the first term in the expanded KL-divergence because it does not depend on the trajectory of the robot $x(t)$. Rather than computing the integral over the exploration space $\mathcal{S}^v$ (partly because of intractability), we approximate the expectation operator as

$$\begin{aligned} D_{\mathrm{KL}}(p\|q) &= -\mathbb{E}_{p(s)}\left[\log q(s)\right] \\ &\approx -\sum_{i=1}^{N} p(s_i) \log q(s_i) \\ &\propto -\sum_{i=1}^{N} p(s_i) \log \int_{t_0}^{t_f} \exp\left[-\frac{1}{2}\|s_i - \bar{x}(t)\|^2_{\Sigma^{-1}}\right] dt, \end{aligned} \tag{4}$$

where N is the number of samples in the search domain drawn uniformly.[4] Through this formulation, we still obtain the benefits of indirectly sampling from the spatial distribution $p(s)$ without having to directly compute derivatives to generate an optimal control signal for the robot. Furthermore, this measure prevents computing the measure from scaling drastically with the number of exploration states. Figure 1 illustrates the resulting reconstruction of the time-averaged statistics of a trajectory $x(t)$ using Definition 3 . The following section uses the KL-divergence ergodic measure and derives a controller which optimizes (4) while directly incorporating learned models and policies.

[4]We can always use importance sampling to interchange which distribution we sample from.

V. KL-E^3: KL-Ergodic Exploration from Equilibrium

In this section, we derive KL-Ergodic Exploration from Equilibrium (KL-E^3), which locally optimizes and improves (4). As an additional constraint, we impose an equilibrium policy and an approximate transition model of the robot's dynamics on the algorithm. By synthesizing KL-E^3 with existing policies that allow the robot to return to an equilibrium state (i.e., local linear quadratic regulators (LQR) controller), we can take advantage of approximate transition models for planning the robot's motion while providing a bound on how unstable the robot can become. We then show how this method is Lyapunov attractive [31], [32], allowing the robot to become unstable so long as we can ensure the robot will eventually return to an equilibrium.

A. Model and Policy Assumptions for Equilibrium:

We assume that we have a robot whose approximate dynamics can be modeled using the continuous time transition model:

$$\begin{aligned} \dot{x}(t) &= f(x(t), \mu(x(t))) \\ &= g(x(t)) + h(x(t))\mu(x(t)) \end{aligned} \tag{5}$$

where $\dot{x}(t) \in \mathbb{R}^n$ is the change of rate of the state $x(t) : \mathbb{R} \to \mathbb{R}^n$ of the robot at time t, $f(x, u) : \mathbb{R}^{n \times m} \to \mathbb{R}^n$ is the (possibly nonlinear) transition model of the robot as a function of state x and control u which we partition into $g(x) : \mathbb{R}^n \to \mathbb{R}^n$, the free unactuated dynamics, and $h(x) : \mathbb{R}^n \to \mathbb{R}^{n \times m}$, the actuated dynamics. In our modeling assumption, we consider a policy $\mu(x) : \mathbb{R}^n \to \mathbb{R}^m$ which provides the control signal to the robotic system such that there exists a continuous Lyapunov function $V(x)$ [32], [33] which has the following conditions:

$$V(0) = 0 \tag{6a}$$

$$\forall x \in \mathcal{B} \backslash \{0\} \quad V(x) > 0 \tag{6b}$$

$$\forall x \in \mathcal{B} \backslash \{0\} \quad \nabla V \cdot f(x, \mu(x)) < 0 \tag{6c}$$

where $\mathcal{B} \subset \mathbb{R}^n$ is a compact and connected set, and

$$\dot{V}(x(t)) = \frac{\partial}{\partial x} V(x) \cdot f(x, u) = \nabla V \cdot f(x, u). \tag{7}$$

Thus, a trajectory $x(t)$ from time $t \in [t_0, t_f]$ subject to (5) and $\mu(x)$ is defined as

$$x(t_f) = x(t_0) + \int_{t_0}^{t_f} f(x(t), \mu(x(t))) dt. \tag{8}$$

For the rest of the paper, we will refer to $\mu(x)$ as an equilibrium policy which is tied to an objective which returns the robot to an equilibrium state. In our prior work [34] we show how one can include any objective into the synthesis of KL-E^3.

B. Synthesizing a Schedule of Exploratory Actions:

Given the assumptions of a known approximate model and an equilibrium policy, our goal is to generate a control signal that augments $\mu(x)$ that minimizes (4) while ensuring that x remains within the compact set $\mathcal{B}$ which will allow the robot to return to an equilibrium state within the time $t \in [t_0, t_f]$.

Our approach starts by quantifying how sensitive (4) is to switching from the policy $\mu(x(t))$ to an arbitrary control vector $\mu_\star(t)$ at any time $\tau \in [t_0, t_f]$ for an infinitesimally small duration of time λ. We will later use this sensitivity to calculate a closed-form solution to the most influential control signal $\mu_\star(t)$.

Proposition 1. *The sensitivity of (4) with respect to the duration time λ, of switching from the policy $\mu(x)$ to an arbitrary control signal $\mu_\star(t)$ at time τ is*

$$\frac{\partial D_{KL}}{\partial \lambda} = \rho(\tau)^\top (f_2 - f_1) \tag{9}$$

where $f_2 = f(x(\tau), \mu_\star(\tau))$ and $f_1 = f(x(\tau), \mu(x(\tau)))$, $\rho(t) \in \mathbb{R}^n$ is the adjoint, or co-state variable which is the solution of the following differential equation

$$\dot{\rho}(t) = \sum_i \frac{p(s_i)}{q(s_i)} \frac{\partial g}{\partial x} - \left(\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u} \frac{\partial \mu}{\partial x} \right)^\top \rho(t) \tag{10}$$

subject to the terminal constraint $\rho(t_f) = \mathbf{0}$, and

$$g = g(s_i \mid x(t)) = \exp\left[-\frac{1}{2} \| s_i - \bar{x}(t) \|^2_{\Sigma^{-1}} \right].$$

Proof. Let us define the trajectory $x(t)$ switching from $\mu(x(\tau)) \to \mu_\star(\tau)$ for a duration of λ as

$$\begin{aligned} x(t_f) = x(t_0) + \int_{t_0}^{\tau} f(x, \mu(x)) dt + \int_{\tau}^{\tau+\lambda} f(x, \mu_\star) dt \\ + \int_{\tau+\lambda}^{t_f} f(x, \mu(x)) dt \end{aligned} \tag{11}$$

where we drop the dependence on time for clarity. Taking the derivative of (4), using (11), with respect to the duration time λ gives us the following expression:

$$\frac{\partial}{\partial \lambda} D_{\text{KL}} = -\sum_i \frac{p(s_i)}{q(s_i)} \int_{\tau+\lambda}^{t_f} \frac{\partial g}{\partial x}^\top \frac{\partial x}{\partial \lambda} dt. \tag{12}$$

Using (11) and Leibniz's rule, we can show that the term $\frac{\partial x}{\partial \lambda}$ is expressed as

$$\frac{\partial x(t)}{\partial \lambda} = (f_2 - f_1) + \int_{\tau+\lambda}^{t} \left(\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u} \frac{\partial \mu}{\partial x} \right)^\top \frac{\partial x(s)}{\partial \lambda} ds \tag{13}$$

where s is a place holder variable for time, $f_2 = f(x(\tau + \lambda), \mu_\star(\tau + \lambda))$ and $f_1 = f(x(\tau + \lambda), \mu(x(\tau + \lambda)))$. Because of the existance of $\frac{\partial x}{\partial \lambda}$ recurring under the integral of (13), we can show that this is a convolution with initial condition $\frac{\partial x(\tau+\lambda)}{\partial \lambda} = f_2 - f_1$. As a result, we can rewrite (13) using a state-transition matrix $\Phi(t, \tau + \lambda)$ [35] as

$$\frac{\partial x(t)}{\partial \lambda} = \Phi(t, \tau + \lambda)(f_2 - f_1). \tag{14}$$

Using (14) in (12) gives the following expression

$$\frac{\partial}{\partial \lambda} D_{\text{KL}} = -\sum_i \frac{p(s_i)}{q(s_i)} \int_{\tau+\lambda}^{t_f} \frac{\partial g}{\partial x}^\top \Phi(t, \tau + \lambda) dt \, (f_2 - f_1). \tag{15}$$

Taking the limit as $\lambda \to 0$ we can set

$$\rho(\tau)^\top = -\sum_i \frac{p(s_i)}{q(s_i)} \int_{\tau}^{t_f} \frac{\partial g}{\partial x}^\top \Phi(t, \tau) dt \tag{16}$$

in (15) which results in

$$\frac{\partial}{\partial \lambda} D_{\text{KL}} = \rho(\tau)^\top (f_2 - f_1). \tag{17}$$

From [1] we can show that $\rho(t)$ can be solved with the following backwards differential equation

$$\dot{\rho}(t) = \sum_i \frac{p(s_i)}{q(s_i)} \frac{\partial g}{\partial x} - \left(\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u} \frac{\partial \mu}{\partial x} \right)^\top \rho(t) \tag{18}$$

with final condition $\rho(t_f) = \mathbf{0}$. □

The sensitivity $\frac{\partial}{\partial \lambda} D_{\text{KL}}$ is known as the mode insertion gradient [1]. Note that the second term in (10) encodes how the dynamics will change subject to the policy $\mu(x)$. We can directly compute the mode insertion gradient for any control $\mu_\star(t)$ that we choose. However, our goal is to find a schedule of $\mu_\star(t)$ which minimizes (4) while still bounded by the equilibrium policy $\mu(x)$. We solve for this augmented control signal by formulate the following optimization problem:

$$\mu_\star(t) = \underset{\mu_\star(t) \forall t \in [t_0, t_f]}{\arg\min} \int_{t_0}^{t_f} \frac{\partial}{\partial \lambda} D_{\text{KL}} \Big|_{\tau=t} + \frac{1}{2} \| \mu_\star(t) - \mu(x(t)) \|^2_{\mathbf{R}} dt \tag{19}$$

where $\mathbf{R} \in \mathbb{R}^{m \times m}$ is a positive definite matrix that penalizes the deviation from the policy $\mu(x)$ and $\frac{\partial}{\partial \lambda} D_{\text{KL}}|_{\tau=t}$ is (9) evaluated at time t.

Proposition 2. *The augmented control signal $\mu_\star(t)$ that minimizes (19) is given by*

$$\mu_\star(t) = -\mathbf{R}^{-1} h(x(t))^\top \rho(t) + \mu(x(t)). \tag{20}$$

Proof. Taking the derivative of (19) with respect to $\mu_\star(t)$ at each instance in time $t \in [t_0, t_f]$ gives

$$\begin{aligned} \int_{t_0}^{t_f} \frac{\partial}{\partial \mu_\star} \left(\frac{\partial}{\partial \lambda} D_{\text{KL}} \Big|_{\tau=t} + \frac{1}{2} \| \mu_\star(t) - \mu(x(t)) \|^2_{\mathbf{R}} \right) dt \\ = \int_{t_0}^{t_f} h(x(t))^\top \rho(t) + \mathbf{R}(\mu_\star(t) - \mu(x(t))) dt \end{aligned} \tag{21}$$

where we expand $f(x, u)$ using (5). Since the expression under the integral in (19) is convex in $\mu_\star(t)$ and is at an optimizer when (21) is equal to $\mathbf{0} \forall t \in [t_0, t_f]$, we set the expression in (21) to zero and solve for $\mu_\star(t)$ at each instant in time giving us

$$\mu_\star(t) = -\mathbf{R}^{-1} h(x(t))^\top \rho(t) + \mu(x(t))$$

which is the schedule of exploratory actions that reduces the objective for time $t \in [t_0, t_f]$ and is bounded by $\mu(x)$. □

In practice, the first term in (20) is calculated and applied to the robot using a true measurement of the state $\hat{x}(t)$ for

the policy $\mu(x)$. We refer to this first term as $\delta\mu_\star(t) = -\mathbf{R}^{-1}h(x(t))^\top\rho(t)$ yielding $\mu_\star(t) = \delta\mu_\star(t) + \mu(\hat{x}(t))$.

Given the derivation of the augmented control signal that can generate ergodic exploratory motions, we verify the following through theoretical analysis in the next section:

- that (20) does in fact reduce (4)
- that (20) imposes a bound on the conditions in (6)
- and that a robotic system subject to (20) has a notion of Lyapunov attractiveness

Theoretical Analysis: We first illustrate that our approach for computing (20) does reduce (4).

Corollary 1. *Let us assume that $\frac{\partial}{\partial\mu}\mathcal{H} \neq 0\ \forall t \in [t_0, t_f]$, where $\mathcal{H}$ is the control Hamiltonian. Then*

$$\frac{\partial}{\partial\lambda}D_{KL} = -\|h(x(t))^\top\rho(t)\|^2_{\mathbf{R}^{-1}} < 0 \tag{22}$$

$\forall t \in [t_0, t_f]$ subject to $\mu_\star(t)$.

Proof. Inserting (20) into (9) and dropping the dependency of time for clarity gives

$$\begin{aligned}\frac{\partial}{\partial\lambda}D_{\text{KL}} &= \rho(t)^\top(f_2 - f_1)\\ &= \rho^\top(g(x) + h(x)\mu_\star - g(x) - h(x)\mu(x))\\ &= \rho^\top(-h(x)\mathbf{R}^{-1}h(x)^\top\rho + h(x)\mu(x) - h(x)\mu(x))\\ &= -\rho^\top h(x)\mathbf{R}^{-1}h(x)^\top\rho\\ &= -\|h(x(t))^\top\rho(t)\|^2_{\mathbf{R}^{-1}} \leq 0.\end{aligned} \tag{23}$$

Thus, $\frac{\partial}{\partial\lambda}D_{\text{KL}}$ is always negative subject to (20). □

For $\lambda > 0$ we can approximate the reduction in D_{KL} as $\Delta D_{\text{KL}} \approx \frac{\partial}{\partial\lambda}D_{\text{KL}}\lambda \leq 0$. Thus, by applying (20), we are generating exploratory motions that minimize the ergodic measure defined by (4).

Our next set of analysis involves searching for a bound on the conditions in (6) when (20) is applied at any time $\tau \in [0, t-\lambda]$ for a duration $\lambda \leq t$.

Theorem 1. *Given the conditions in (6) for a policy $\mu(x)$, then $V(x^\tau_\lambda(t)) - V(x(t)) \leq \lambda\beta$, where $x^\tau_\lambda(t)$ is the solution to (11) subject to (20) for $\tau \in [0, t-\lambda]$, $\lambda \leq t$, and*

$$\beta = \sup_{t\in[\tau,\tau+\lambda]} -\nabla V \cdot h(x(t))\mathbf{R}^{-1}h(x(t))^\top\rho(t). \tag{24}$$

Proof. Writing the integral form of the Lyapunov function switching between $\mu(x(t))$ and $\mu_\star(t)$ at time τ for a duration of time λ starting at $x(0)$ can be written as

$$\begin{aligned}V(x^\tau_\lambda(t)) = V(x(0)) &+ \int_0^\tau \nabla V \cdot f(x(s), \mu(x(s)))ds\\ &+ \int_\tau^{\tau+\lambda} \nabla V \cdot f(x(s), \mu_\star(s))ds\\ &+ \int_{\tau+\lambda}^t \nabla V \cdot f(x(s), \mu(x(s)))ds\end{aligned} \tag{25}$$

where s is a place holder for time. Expanding $f(x,u)$ to $g(x) + h(x)u$ and using (20) we can show the following identity:

$$\begin{aligned}\nabla V \cdot f(x, \mu_\star) &= \nabla V \cdot g(x) + \nabla V \cdot h(x)\mu_\star\\ &= \nabla V \cdot g(x) + \nabla V \cdot h(x)\mu(x)\\ &\quad - \nabla V \cdot h(x)\mathbf{R}^{-1}h(x)^\top\rho\\ &= \nabla V \cdot f(x, \mu(x)) - \nabla V \cdot h(x)\mathbf{R}^{-1}h(x)^\top\rho\end{aligned} \tag{26}$$

Using (26) in (25), we can show that

$$\begin{aligned}V(x^\tau_\lambda(t)) &= V(x(0)) + \int_0^t \nabla V \cdot f(x(s), \mu(x(s)))ds\\ &\quad - \int_\tau^{\tau+\lambda} \nabla V \cdot h(x(s))\mathbf{R}^{-1}h(x(s))^\top\rho(s)ds\\ &= V(x(t))\\ &\quad - \int_\tau^{\tau+\lambda} \nabla V \cdot h(x(s))\mathbf{R}^{-1}h(x(s))^\top\rho(s)ds\end{aligned} \tag{27}$$

where $x(t)$ is given by (8).

Letting the largest value of $\nabla V \cdot h(x(t))\mathbf{R}^{-1}h(x(t))^\top\rho(t)$ be given by

$$\beta = \sup_{s\in[\tau,\tau+\lambda]} -\nabla V \cdot h(x(s))\mathbf{R}^{-1}h(x(s))^\top\rho(s) > 0, \tag{28}$$

we can approximate (27) as

$$\begin{aligned}V(x^\tau_\lambda(t)) &= V(x(t)) - \int_\tau^{\tau+\lambda} \nabla V \cdot h(x(s))\mathbf{R}^{-1}h(x(s))^\top\rho(s)ds\\ &\leq V(x(t)) + \beta\lambda.\end{aligned}$$

Subtracting both side by $V(x(t))$ gives the upper bound

$$V(x^\tau_\lambda(t)) - V(x(t)) \leq \beta\lambda \tag{29}$$

which quantifies how much (20) deviates from the equilibrium conditions in (6). □

We can choose any time $\tau \in [0, t-\lambda]$ to apply $\mu_\star(t)$ and provide an upper bound quantifying the change of the Lyaponov function described in (6) by fixing the maximum value of λ during active exploration. In addition, we can tune $\mu_\star(t)$ using the regularization value $\mathbf{R}$ such that as $\|\mathbf{R}\| \to \infty$, $\beta \to 0$ and $\mu_\star(t) \to \mu(x(t))$.

Given this bound, we can guarantee Lyapunov attractiveness [8], where the robot is not Lyapunov stable, but rather there exists a time t such that the system (8) is guaranteed to return to a region of attraction where the system can be guided towards a stable equilibrium state x_0.

Definition 4. *A robotic system defined by (5) is Lyapunov attractive if at some time t, the trajectory of the system $x(t) \in \mathcal{C}(t) \subset \mathcal{B}$ where $\mathcal{C}(t) = \{x(t)|V(x) \leq \beta^\star, \nabla V \cdot f(x(t), \mu(x(t))) < 0\}$, $\beta^\star > 0$ is the maximum level set of $V(x)$ where $\nabla V \cdot f(x, \mu(x)) < 0$, and $\lim_{t\to\infty} x(t) \to x_0$ such that x_0 is an equilibrium state.*

Theorem 2. *Given the schedule of exploratory actions (20) $\forall t \in [\tau, \tau+\lambda]$, a robotic system governed by (5) is Lyapunov attractive such that $\lim_{t\to\infty} x^\tau_\lambda(t) \to x_0$.*

Proof. Using Theorem 1, the integral form of the Lyapunov function (25), and the identity (26), we can write

$$V(x_\lambda^\tau(t)) = V(x(0)) + \int_0^t \nabla V \cdot f(x(s), \mu(x(s)))ds - \int_\tau^{\tau+\lambda} \nabla V \cdot h(x(s))\mathbf{R}^{-1}h(x(s))^\top \rho(s)ds$$

$$\leq V(x(0)) - \gamma t + \beta\lambda < \beta^\star,$$

where

$$-\gamma = \sup_{s\in[0,t]} \nabla V \cdot f(x(s), \mu(x(s))) < 0. \tag{30}$$

Since λ is fixed and β can be tuned by the matrix weight $\mathbf{R}$, we can choose a t such that $\gamma t \gg \beta\lambda$. Thus, $\lim_{t\to\infty} V(x_\lambda^\tau(t)) \to V(x_0)$ and $\lim_{t\to\infty} x_\lambda^\tau(t) \to x_0$, implying Lyapunov attractiveness, where $V(x_0)$ is the minimum of the Lyapunov function at the equilibrium state x_0.

□

Proving Lyapunov attractiveness allows us to make the claim that a robot will return to a region where $\dot{V}(x) < 0$ subject to the policy $\mu(x)$. This enables the robot to actively explore states which would not naturally have a safe recovery. Moreover, this analysis shows that we can choose the value of λ and $\mathbf{R}$ when calculating $\mu_\star(t)$ such that attractiveness always holds, giving us an algorithm that is safe for active learning.

So far, we have shown that (20) is a method that generates approximate ergodic exploration from equilibrium policies. We prove that this approach does reduce (4) and show the ability to quantify and bound how much the active exploration process will deviate the robotic system from equilibrium. Last, it is shown that generating data for learning does not require constant guaranteed Lyapunov stability of the robotic system, but instead introduce the notion of attractiveness where we allow the robot to explore the physical realm so long as the time to explore is finite and the magnitude of the exploratory actions is restrained. In the following section, we extend our previous work in [34] by further approximating the time averaged-statistics so that computing the adjoint variable can be done more efficiently.

C. KL-E^3 for Efficient Planning and Exploration

We extend our initial work by providing a further approximation to computing the time-averaged statistics which improves the computation time of our implementation. Taking note of (4), we can see that we have to evaluate $q(s_i)$ at each sample point s_i where $q(s) = q(s|x(t))$ has to evaluate the stored trajectory at each time. In real robot experiments, often the underlying spatial statistics $p(s)$ change significantly over time. In addition, most robot experiments require replanning which results in lost information over repeated iterations. Thus, rather than trying to compute the whole time averaged trajectory in Definition 3, we opt to approximate the distribution by applying Jensen's inequality:

$$q(s|x(t)) \propto \int_{t_0}^{t_f} \exp\left[-\frac{1}{2}\|s-\bar{x}(t)\|^2_{\Sigma^{-1}}\right] dt \geq \exp\left(-\frac{1}{2}\int_{t_0}^{t_f} \|s-\bar{x}(t)\|^2_{\Sigma^{-1}} dt\right). \tag{31}$$

Using this expression in (4), we can write

$$\begin{aligned} D_{\text{KL}} &\propto -\int_{\mathcal{S}^v} p(s)\log q(s)ds \\ &= -\int_{\mathcal{S}^v} p(s)\log\left(\exp\left(-\frac{1}{2}\int_{t_0}^{t_f}\|s-\bar{x}(t)\|^2_{\Sigma^{-1}}dt\right)\right)ds \\ &\propto \int_{\mathcal{S}^v} p(s)\left(\int_{t_0}^{t_f}\|s-\bar{x}(t)\|^2_{\Sigma^{-1}}dt\right)ds \\ &\approx \sum_i^N p(s_i)\int_{t_0}^{t_f}\|s_i-\bar{x}(t)\|^2_{\Sigma^{-1}}dt. \end{aligned} \tag{32}$$

Following the results to compute (9), we can show that (20) remains the same where the only modification is in the adjoint differential equation where

$$\dot{\rho}(t) = -\sum_i p(s_i)\frac{\partial g}{\partial x} - \left(\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial \mu}{\partial x}\right)^\top \rho(t) \tag{33}$$

such that $g = g(s|x) = \|s-x\|^2_{\Sigma^{-1}}$. This formulation has no need to compute $q(s)$ and instead just evaluate $p(s)$ at sampled points. We reserve using this implementation for robotic systems that are of high dimensional space or when calculating the derivatives can be costly due to over-parameterization (i.e., multi-layer networks). Note that all the theoretical analysis still holds because the fundamental theory relies on the construction through hybrid systems theory rather than the KL-divergence itself. The downside to this approach is that one now loses the ability to generate consistent ergodic exploratory movements. The effect can be seen in Figure 1(e) where the trajectory is approximated by a wide Gaussian—rather than the bi-model distribution found in Fig. 1(d). However, for non-stationary $p(s)$, having exact ergodic behavior is not necessary and such approximations at the time-averaged distribution level are sufficient.

In the following subsection, we provide base algorithm and implementation details for KL-E^3 and present variations based on the learning goals in VI in the Appendix.

D. Algorithm Implementation

In this section, we provide an outline of a base implementation of KL-E^3 in Algorithm 1. We also define some variables which were not previously mentioned in the derivation of KL-E^3 and provide further implementation detail.

There exist many ways one could use (20). For instance, it is possible to simulate a dynamical system for some time horizon t_H and apply (20) in a trajectory optimization setting. Another way is to repeatedly generate trajectory plans at each instance and apply the first action in a model-based predictive control (MPC) manner. We found that the choice of τ and λ can determine how one will apply (20). That is, given some time

t_i, if $\tau = t_i$ and $\lambda = t_H$, we recover the trajectory optimization formulation whereas $\tau = t_i$, $\lambda = dt$, where dt is the time step, then the MPC formulation is recovered.[5] Rather than focusing on incremental variations, we focus on the general structure of the underlying algorithm when combined with a learning task.

We fist assume that we have an approximate transition model $f(x, u)$ and an equilibrium policy $\mu(x)$. A simulation time horizon t_H and a time step dt is specified where true robot measurements of state are given by $\hat{x}(t)$ and the simulated states are $x(t)$. Last, a spatial distribution $p(s)$ is initialized (usually uniform to start), and an empty data set $\mathcal{D} = \{\hat{x}(t_j), y(t_j)\}_j$ is initialized where $y(t)$ are measurements. In the following sections, we provide examples for constructing and updating $p(s)$ given various learning tasks.

Provided the initial items, the algorithm first samples the robot's state $\hat{x}(t_i)$ at the current time t_i. Using the transition model and policy, the next $t_i + t_H$ states $x(t) \forall t \in [t_i, t_i + t_H]$ are simulated. A set of N, samples $(s_1, s_2, \ldots, s_N)$ are generated and used to compute $p(s), q(s)$. The adjoint variable is then backwards simulated from $t = t_i + t_H \rightarrow t_i$ and is used to compute $\delta\mu_\star(t)$. We ensure robot safety by applying $\delta\mu_\star(t) + \mu(\hat{x}(t))$ with real measurements of the robot's state. Data is then collected and appended to $\mathcal{D}$ and used to update $p(s)$. Any additional steps are specified by the learning task. The pseudo-code for this description is provided in Algorithm 1 .

Algorithm 1 KL-E^3 Base Algorithm

init: approximate transition model $f(x, u)$, initial true state $\hat{x}(0)$, equilibrium policy $\mu(x)$, spatial distribution $p(s)$, simulation time horizon t_H, time step dt. data set $\mathcal{D}$, $i = 0$
while task not done **do**
  set $x(t_i) = \hat{x}(t_i)$
  ▷ simulation loop
  for $\tau_i \in [t_i, \ldots, t_i + t_H]$ **do**
    ▷ forward predict states using any
    ▷ integration method (Euler shown)
    $x(\tau_{i+1}) = x(\tau_i) + f(x(\tau_i), \mu(x(\tau_i)))dt$
  ▷ backwards integrate choosing $\dot{\rho}(t)$
  ▷ set the terminal condition
  generate N samples of s_i uniformly within $\mathcal{S}^v$
  $\rho(t_i + t_H) = \mathbf{0}$
  for $\tau_i \in [t_H + t_i, \ldots, t_i]$ **do**
    $\rho(\tau_{i-1}) = \rho(\tau_i) - \dot{\rho}(\tau_i)dt$
    ▷ since $x(t)$ is simulated, we return
    ▷ just the first term of (20)
    ▷ and calculate $\mu(x)$ online
    $\delta\mu_\star(\tau_{i-1}) = -\mathbf{R}^{-1}h(x(\tau_{i-1}))^\top \rho(\tau_{i-1})$
  ▷ apply to real robot
  chose $\tau \in [t_i, t_i + t_H]$ and $\lambda \leq t_H$ or use line search [36]
  for $t \in [t_i, t_{i+1}]$ **do**
    if $t \in [\tau, \tau + \lambda]$ **then**
      apply $\mu_\star(t) = \delta\mu_\star(t) + \mu(\hat{x}(t))$
    else
      apply $\mu(x(t))$
    if time to sample **then**
      measure true state $\hat{x}(t)$ and measurements $y(t)$
      append to data set $\mathcal{D} \leftarrow \{\hat{x}(t), y(t)\}$
  update $p(s)$ given $\mathcal{D}$ ▷ task specific
  update $f(x, u), \mu(x)$ ▷ if needed
  update learning task
  $i \leftarrow i + 1$

VI. Example Learning Goals

In this section, our goal is to use KL-E^3 for improving example methods for learning. In particular, we seek to show that our method can improve Bayesian optimization, transition model learning (also known as dynamics model learning or system identification), and off-policy robot skill learning. In each subsection, we provide an overview of the learning goal and define the spatial distribution $p(s)$ used in our method. In addition, we show the following:

- that our method is capable of improving the learning process through exploration
- that our method does not violate equilibrium policies and destabilize the robot
- and that our method efficiently explores through exploiting the dynamics of a robot and the underlying spatial distribution.

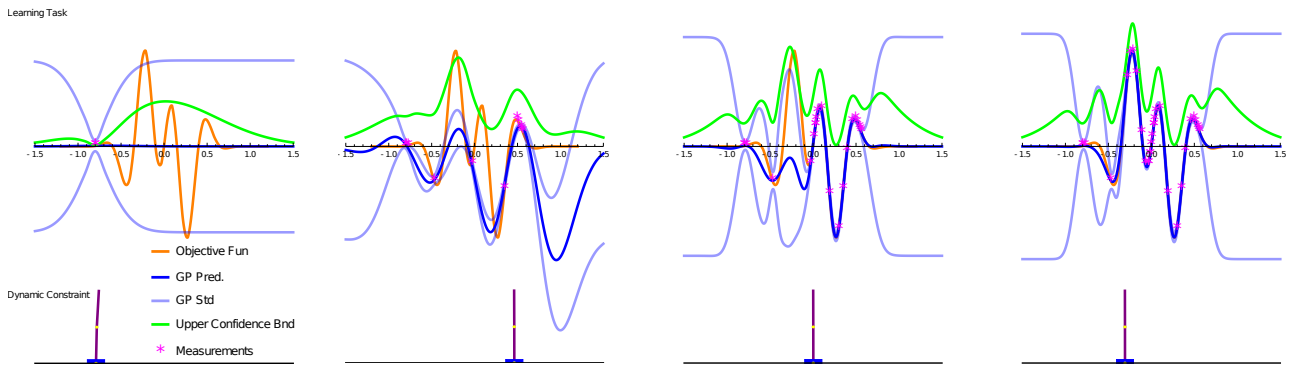
For each example, we provide implementation, including parameters used, in the appendix.

A. Bayesian Optimization

In our first example, we explore KL-E^3 for Bayesian optimization using a cart double pendulum system [37] that needs to maintain itself at the upright equilibrium. Bayesian optimization is a probabilistic approach for optimizing objective functions $\phi(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ that are either expensive to evaluate or are highly nonlinear. A probabilistic model (often a Gaussian process) of the objective function is built from sampled data $x_k \in \mathbb{R}^n$ and the posterior of the model is used to construct an acquisition function [22]. The acquisition function maps the sample space x to a value which indicates the utility of the sample (in other words, how likely is the sample to provide information about the objective given the previous sample in x). The acquisition function is often simpler and easier to calculate for selecting where to sample next rather than the objective function itself. Assuming one can freely sample the space x, Bayesian optimization takes a sample based on the acquisition function and a posterior is computed. The process then repeats until some terminal number of evaluations of the objective or the optimizer is reached. We provide pseudo-code for Bayesian optimization in the Appendix, Alg 2.

In many examples of Bayesian optimization, the assumption is that the learning algorithm can freely sample anywhere in the sample space $x \in \mathbb{R}^n$; however, this is not always true.

[5] One can also automate choosing τ and λ using a line search [36].



(a) $t = 0$ (b) $t = 10$ (c) $t = 20$ (d) $t = 30$

Fig. 2: Time series snap-shots of cart double pendulum actively sampling and estimating the objective function (orange). The uncertainty (light blue) calculated from the collected data set drives the exploratory motion of the cart double pendulum while our method ensures that the cart double pendulum is maintained in its upright equilibrium state.

Consider an example where a robot must collect a sample from a Bayesian optimization step where the search space of this sample intersects the state-space of the robot itself. The robot is constrained by its dynamics in terms of how it can sample the objective. Thus, the Bayesian optimization step becomes a constrained optimization problem where the goal is to reach the optimal value of the acquisition function subject to the dynamic constraints of the robot. Furthermore, assume that the motion of the robot is restricted to maintain the robot at an equilibrium (such as maintaining the inverted equilibrium of the cart double pendulum). The problem statement is then to enable a robot to execute a sample step of Bayesian optimization by taking into account the constraints of the robot. We use this example to emphasize the effectiveness of our method for exploiting the local dynamic information using a cart double pendulum where the equilibrium state is at the upright inverted state and a policy maintains the double pendulum upright.

Here, we use a Gaussian process with the radial basis function (RBF) to build a model of the objective function shown in Fig. 2. The upper confidence bound (UCB) [22] of the Gaussian process posterior is used as the acquisition function and is defined as

$$\text{UCB}(x) = \bar{\mu}(x) + \kappa\sigma(x) \tag{34}$$

where $\kappa > 0$. We augment Alg 1 for Bayesian optimization by setting the UCB acquisition function as the target distribution which we define through the Boltzmann softmax function [38], [39]:

$$p(s) = \frac{\exp(c\text{UCB}(s))}{\int_{\mathcal{S}^v} \exp(c\text{UCB}(\bar{s}))d\bar{s}} \tag{35}$$

where $c > 0$ is a scaling constant. Note that the denominator is approximated as a sum over the samples that our method generates. An approximate linear model of the cart double pendulum dynamics centered around the unstable equilibrium is used along with an LQR policy that maintains the double pendulum upright. We provide brief pseudo-code of the base Algorithm 1 in the Appendix Alg. 3.

We first illustrate that our method generates ergodic exploration through an execution of our method for Bayesian optimization in Figure 2. Here, the time-series evolution of KL-E^3 is shown to sample proportional to the acquisition function. As a result, our method generates samples near each of the peaks of the objective function. Furthermore, we can see that our method is exploiting the dynamics as well as the equilibrium policy, maintaining Lyapunov attractiveness with respect to the inverted equilibrium (we will later discuss numerical results in Fig 4).

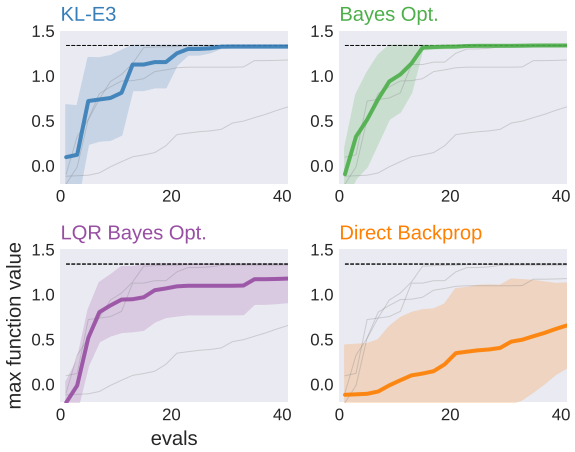


Fig. 3: Comparison of KL-E^3 against Bayesian Optimization without dynamic constraint, LQR-Bayesian optimization, and direct maximization of the acquisition function through gradient propagation of the cart double pendulum approximate dynamics in determining the maximum value of the objective function through exploration. Our method is able to perform as well as Bayesian optimization directly sampling the exploration space while performing better than the naive LQR-Bayesian optimization. Dashed black line indicates the maximum value of the function.

Next, our method is compared against three variants of Bayesian optimization: the first is Bayesian optimization with no dynamics constraint (i.e., no robot is used); second, a linear quadratic regulator (LQR) variation of Bayesian optimization

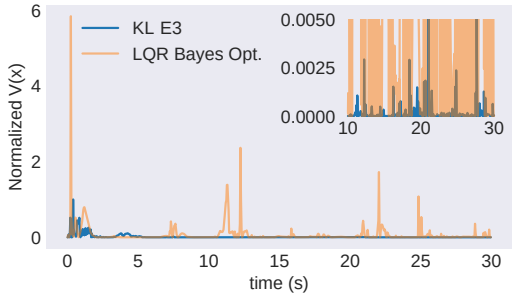


Fig. 4: Normalized Lyapunov function for the cart double pendulum with upright equilibrium. Our method (shown in blue) is roughly 6 times more stable than LQR-Bayesian optimization. The large initial values indicate the sweeping shown in Fig. 2(a) when the cart double pendulum moves across the search space. Subsequent application of the exploratory motion refine the exploration process. The Lyapunov attractiveness property is enforced through automatic switching of the exploration process.

where the maximum of the acquisition function is used as a target for an LQR controller; and last a direct maximization of the acquisition using the stabilizing equilibrium policy (see [40] for detail) is used. A total of 10 trials for each method are collected with the agent starting at the same location uniformly sampled between -0.8 and 0.8 throughout the sample space. In Fig. 3 we show that our method not only performs comparably to Bayesian optimization without dynamic constraints[6], but outperforms both LQR and direct maximization variants of Bayesian optimization. Because LQR-Bayes method does not take into account dynamic coverage, and instead focuses on reaching the next sample point, the dynamics of the robot often do not have sufficient time to stabilize which leads to higher variance of the learning objective. We can see this in Fig. 4 where we plot a Lyapunov function for the cart double pendulum [37] at the upright unstable equilibrium. Specifically, our method is roughly 6 times more stable at the start of the exploration compared to the LQR variant of Bayesian optimization. Lyapunov attractiveness is further illustrated in Fig. 4 as time progresses and each exploratory motion is closer to the equilibrium. Last, directly optimizing the highly nonlinear acquisition function often leads to local optima, yielding poor performance in the learning goal. This can be seen with the performance of directly optimizing UCB using the cart double pendulum approximate dynamics in Fig. 3) where the cart double pendulum would often find and settle at a local optima.

In this example, the cart double pendulum only needed to explore the cart position domain to find the maximum of the objective function. The following example illustrates a more dynamic learning goal where the robot needs to generate a stochastic model of its own dynamics through exploration within the state-space.

[6]This may change if the dynamics of the robot are slower or the exploration space is sufficiently larger. Note that the other methods would also be equally affected.

B. Stochastic Transition Model Learning

In this next example KL-E^3 is used to collect data for learning a stochastic transition model of a quadcopter [41] dynamical system by exploring the state-space of the quadcopter while remaining at a stable hover. Our goal is to show that our method can efficiently and effectively explore the state-space of the quadcopter (including body linear and angular velocities) in order to generate data for learning a transition model of the quadcopter for model-based control. In addition, we show that the exploratory motions improve the quality of data generated for learning while exploiting and respecting the stable hover equilibrium in a single execution of the robotic system [40].

An LQR policy is used to keep the vehicle hovering while a local linear model (centered around the hover) is used for planning exploratory motions. The stochastic model of the quadcopter is of the form [42]

$$dx \sim \mathcal{N}(f_\theta(x,u), \sigma_\theta(x)) \tag{36}$$

where $\mathcal{N}$ is a normal distribution with mean f_θ, and variance $\sigma_\theta(x)$, and the change in the state is given by $dx \in \mathbb{R}^n$. Here, $f(x,u;\theta) = f_\theta(x,u) : \mathbb{R}^{n\times m} \to \mathbb{R}^n$ specifies a neural-network model of the dynamics and $\sigma(x;\theta) = \sigma_\theta(x)$ is a diagonal Gaussian $\sigma_\theta(x) : \mathbb{R}^n \to \mathbb{R}^n$ which defines the uncertainty of the transition model at state x all parameterized by the parameters θ.

We use KL-E^3 to enable the quadcopter to explore with respect to the variance of the model (that is, exploration in the state-space is generated based on how uncertain the transition model is at that state). In a similar manner as done in the previous subsection, we use a Boltzmann softmax function to create the distribution

$$p(s) = \frac{\exp(c\sigma_\theta(s))}{\int_{\mathcal{S}^v} \exp(c\sigma_\theta(\bar{s}))d\bar{s}}. \tag{37}$$

A more complex target distribution can be built (see [40], [43]), however; due to the over-parameterization of the neural-network model, using such methods would require significant computation.

The stochastic model is optimized by maximizing the log likelihood of the model using the likelihood function

$$\mathcal{L} = \mathcal{N}(dx \mid f_\theta(x,u), \sigma_\theta(x)) \tag{38}$$

where updates to the parameters θ are defined through the gradient of the log likelihood function:

$$\theta \leftarrow \theta + \alpha \sum_k \nabla_\theta \log \mathcal{L}. \tag{39}$$

Here, a batch of K measurements $\{\hat{x}_k, d\hat{x}_k, u_k\}_{k=1}^K$ are uniformly sampled from the data buffer $\mathcal{D}$ where the subscript k denotes the k^{th} time. A variation of Alg. 1 for model learning is provided in the Appendix in Algorithm 4.

We compare our method against time-correlated Ornstein-Uhlenbeck (OU) noise [44], uniform and normally distributed random noise at different noise levels, and using a nonlinear dynamics variant of the information maximizing method in [40], [43] which directly maximizes the variance of the model subject to the equilibrium policy. Each simulation is

Method	Average Power Loss	Average $\|u\|$
KL-E^3	0.16 +- 0.0130	0.68 +- 0.0043
Inf. Max*	0.59 +- 0.0463	1.32 +- 0.3075
Normal 0.1*	1.41 +- 0.0121	0.72 +- 0.0016
OU 0.3	2.73 +- 0.0228	1.17 +- 0.0152
OU 0.1	0.97 +- 0.0096	0.73 +- 0.0033
OU 0.01*	0.10 +- 0.0007	0.67 +- 0.0002
Uniform 0.1*	0.84 +- 0.0090	0.69 +- 0.0004

TABLE I: Comparison of our method against various methods for state-space exploration using a quadcopter. Each method uses the same base stabilization policy which maintains hover height and is instantiated and run once for 1200 time steps. Data from a single execution is used to generate a neural network dynamics model. This is repeated 20 times to estimate the performance of each method. Methods with (*) were unable to generate a dynamics model that completed the tracking objective.

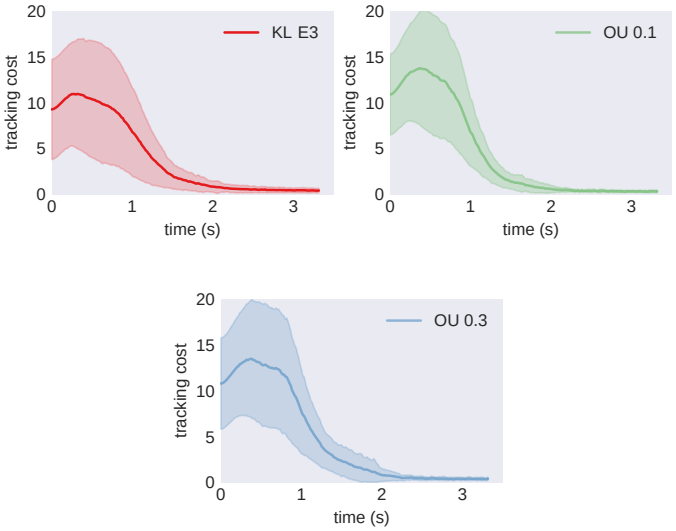


Fig. 5: Learned quadcopter model evaluations on a model-based tracking objective. Our method is able to generate a model that performs comparably to OU noise at 0.1 and 0.3 noise levels while using less energy through dynamic exploration.

run using the LQR controller as a equilibrium policy for a single execution of the robot (no episodic resets) for 1200 time steps. During this time, data is collected and stored in the buffer $\mathcal{D}$. Our method and the information maximizing method use the data in the stored buffer to update the variance $\sigma_\theta(x)$, guiding the exploration process, however; for evaluation of the transition model, we separately learn a model using the data that has been collected as a gauge for the utility of the collected data for each method. A stochastic model is learned by sampling a batch of 200 measurements offline from the buffer using 2000 gradient iterations. The model is evaluated for target tracking using stochastic model-based control [45] over a set of uniformly randomly generating target locations $\mathcal{U}(-2, 2) \in \mathbb{R}^3$.

We first illustrate that our method is more energetically efficient compared to other methods in Table I. Here, energy is calculated using the resulting thrust of the quadcopter and we show the average commanded action u over the execution of the quadrotor in time. Our method is shown to be more energetically efficient (due to the direct exploitation of the equilibrium policy and the ergodic exploration process). Furthermore, our method is able to generate measurements in the state-space that can learn a descriptive stochastic model of the dynamics for model-based tracking control (methods that could not learn a model for tracking control are indicated with a [*]). The resulting methods that could generate a model were comparable to our method (see Fig. 5), however; our method is able to directly target the regions of uncertainty (see Fig. 6) through dynamic exploration allowing the quadcopter to use less energy (and more directed exploratory actions).

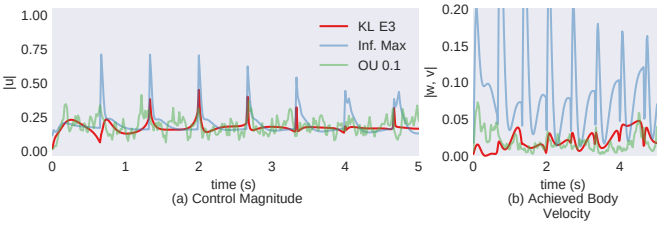


Fig. 6: Control signal $\|u(t)\|$ and resulting body linear and angular velocities ω, v for the quadcopter system using our method, information maximization, and OU noise with 0.1 maximum noise level. Our method generates smoother control signals while exploring the necessary regions of the state-space without destabilizing the system.

Our last example illustrates how method can be used to aide exploration for off-policy robot skill learning methods by viewing the learned skill as an equilibrium policy.

C. Robot Skill Learning

In our last example, we explore KL-E^3 for improving robot skill learning (here we consider off-policy reinforcement learning). Our goal is to show that we can view a robot skill as being in equilibrium (using a feedback policy) where our method can explore within the vicinity of the robot skill in an intentional manner, improving the learning process.

In many examples of robot skill learning, a common mode of failure is that the resulting learned skill is highly dependent on the quality of the distribution of data generated that is used for learning. Typically, these methods use the current iteration of the learned skill (which is often referred to as a policy) with added noise (or have a stochastic policy) to explore the action space. Often the added noise is insufficient towards gaining informative experience which improves the quality of the policy. Here, we show that our method can improve robot skill learning by generating dynamic coverage and exploration around the learned skill, reducing the likelihood of suboptimal solutions, and improving the efficiency of these methods.

We use deep deterministic policy gradient (DDPG) [19] as our choice of off-policy method. Given a set of data, DDPG calculates a Q-function defined as

$$Q(x, u) = \mathbb{E}\left[r(x, u) + \gamma Q(x', \mu(x'))\right] \tag{40}$$

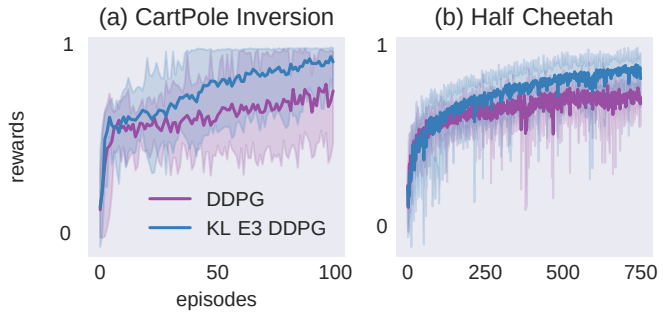


Fig. 7: Comparison of KL-E^3 enhanced DDPG against DDPG using the cart pole inversion and the half cheetah running tasks. KL-E^3 provides a more informative distribution of data which assists the learning process, improves the overall performance, and achieves better performance faster for DDPG.

where $r(x,u) : \mathbb{R}^{n\times m} \to \mathbb{R}$ is a reward function, x' is the next state subject to the control u, the expectation $\mathbb{E}$ is taken with respect to the states, $0 > \gamma > 1$ is known as a discounting factor [39], and the function $Q(x,u) : \mathbb{R}^{n\times m} \to \mathbb{R}$ maps the utility of a state and how the action at that state will perform in the next state given the policy $\mu(x)$. DDPG simultaneously learns $Q(s)$ and a policy $\mu(x)$ by sampling from a set of collected states, actions, rewards, and their resulting next state. We refer the reader to the pseudo-code of DDPG in [19].

Our method uses the learned $Q(x,u)$ and $\mu(x)$ as the target distribution and the equilibrium policy respectively. We modify the Q function such that it becomes a distribution using a Boltzmann softmax

$$p(s) = \frac{\exp(cQ(s))}{\int_{\mathcal{S}^v} \exp(cQ(\bar{s}))d\bar{s}} \tag{41}$$

where $s \in \mathbb{R}^{n+m}$ includes both states and actions. This form of Equation (41) has been used previously for inverse reinforcement learning [46], [47]. Here, Eq. (41) is used as a guide for the ergodic exploration where our exploration is centered around the learned policy and the utility of the learned skill (Q-function). Since most reinforcement learning deals with large state-spaces, we use the approximation to the time-averaged statistics in (31) to improve the computational efficiency of our algorithm. A parameterized dynamics model is built using the first 200 points of each simulation (see Appendix for more detail) and updated as each trial continues. OU noise is used for exploration in the DDPG comparison with the same parameters shown in [19]. We provide a pseudo-code of a KL-E^3 enhanced DDPG in the Appendix Alg. 5.

Our method is tested on the cart pole inversion and the half-cheetah running task (see Figure 7 for results). For both robotic systems, KL-E^3 is shown to improve the overall learning process, making learning a complex robot skill more sample efficient. Specifically, inverting the cart pole starts to occur within 50 episodes and the half cheetah begins generating running gaits within 250 episodes of the half cheetah (each episode consists of 200 time steps of each environment). In contrast, DDPG alone generates suboptimal running (as shown in (https://sites.google.com/view/kle3/home)) and unstable cart inversion attempts. Because our method is able to explore within the vicinity of the learned skill in an intentional, ergodic manner, it is able to quickly learn skills and improve the overall quality of the exploration.

VII. Conclusion

We present KL-E^3, a method which is shown to enable robots to actively generate informative data for various learning goals from equilibrium policies. Our method synthesizes ergodic coverage using a KL-divergence measure which generates data through exploiting dynamic movement proportional to the utility of the data. We show that hybrid systems theory can be used to synthesize a schedule of exploration actions that can incorporate learned policies and models in a systematic manner. Last, we present examples that illustrate the effectiveness of our method for collecting and generating data in an ergodic manner and provide theoretical analysis which bounds our method through Lyapunov attractiveness.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grants CNS 1837515. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the aforementioned institutions.

Appendix A
Algorithmic Details

This appendix provides additional details for each learning goal presented in Section VI. This includes pseudo-code for each method and parameters to implement our examples (see Table. II). We provide videos of each example and demo code in (https://sites.google.com/view/kle3/home).

Algorithm 2 Bayesian Optimization

init: Gaussian prior on objective ϕ, data set $\mathcal{D}$, $i = 0$
while task not done **do**
    update posterior distribution on ϕ using $\mathcal{D}$
    build acquisition function using current posterior on ϕ
    active learner finds the maximum x_i of the acquisition function
    active learner samples $y_i = \phi(x_i)$
    $i \leftarrow i + 1$
    set $x(t_i) = \hat{x}(t_i)$
return max y_i of $\phi(x)$ and argmax x_i

Example	t_H	λ	$f(x,u)$	$\mu(x)$	$\mathbf{R}$	Σ	N samples
Cart Double Pendulum Bayes. Opt.	0.2 s	line search $\lambda < t_H$	local linear model at inverted pose	LQR stabilizing policy	0.1	0.1	20
Quadcopter Model Learning	0.6 s	$\lambda = t_H$	local linear model at hoverheight	LQR hovering policy	$0.5\mathbf{I}$	$0.1\mathbf{I}$	100
DDPG Cart pole swingup	0.1 s	$\lambda = 0.02$ s	neural-net model $\dot{x} = f(x,u;\theta)$ learned from data	Learned swingup skill	$0.01 * 0.99^t$	$0.1\mathbf{I}$	20
DDPG Half Cheetah running	0.03 s	$\lambda = 0.0165s$	neural-net model $\dot{x} = f(x,u;\theta)$ learned from data	Learned running skill	$0.01 * 0.99^t$	$0.1\mathbf{I}$	50

TABLE II: Parameters used for each method presented in Section VI. $\mathbf{I}$ indicates an identity matrix of size $\mathbb{R}^{m\times m}$ for $\mathbf{R}$, $\mathbb{R}^{n\times n}$ for the quadcopter model learning, and $\mathbb{R}^{v\times v}$ where $v = n+m$ used in both DDPG examples. Neural net model parameters θ are learned by minimizing the error $\|\dot{x} - f(x,u;\theta)\|^2$ over a subset of K data points where $\dot{x} \approx (x(t_1) - x(t_0))/dt$ and dt is the time step of the system. A time-decaying $\mathbb{R}$ is used for both DDPG examples so that the largest exploring occurs earlier on in the episode to better assist the skill learning.

Algorithm 3 KL-E^3 for Bayesian Optimization

init: see Alg. 1 and Alg. 2
while task not done **do**
  set $x(t_i) = \hat{x}(t_i)$
  ▷ simulation loop (see Alg. 1 lines 4-18)
  get $\delta\mu(t)$ from simulation
  ▷ apply to real robot (see Alg. 1 lines 20-28)
  chose $\tau \in [t_i, t_i + t_H]$ and $\lambda \leq t_H$
  ▷
  for $t \in [t_i, t_{i+1}]$ **do**
    if $t \in [\tau, \tau+\lambda]$ **then**
      apply $\mu_\star(t) = \delta\mu_\star(t) + \mu(\hat{x}(t))$
    else
      apply $\mu(x(t))$
    if time to sample **then**
      measure true state $\hat{x}(t)$ and $y(t) = \phi(\hat{x}(t))$
      append to data set $\mathcal{D} \leftarrow \{\hat{x}(t), y(t)\}$
  update posterior on ϕ given $\mathcal{D}$
  update $p(s)$ from posterior
  $i \leftarrow i+1$

Algorithm 4 KL-E^3 for Model Learning

init: see Alg. 1, generate initial parameter θ^0 for model (36)
while task not done **do**
  set $x(t_i) = \hat{x}(t_i)$
  ▷ simulation loop (see Alg. 1 lines 4-18)
  get $\delta\mu(t)$ from simulation
  ▷ apply to real robot
  chose $\tau \in [t_i, t_i + t_H]$ and $\lambda \leq t_H$
  for $t \in [t_i, t_{i+1}]$ **do**
    if $t \in [\tau, \tau+\lambda]$ **then**
      apply $\mu_\star(t) = \delta\mu_\star(t) + \mu(\hat{x}(t))$
    else
      apply $\mu(x(t))$
    if time to sample **then**
      measure state $\hat{x}(t)$, change in state $d\hat{x}(t)$, and applied control $u(t)$
      append to data set $\mathcal{D} \leftarrow \{\hat{x}(t), d\hat{x}(t), u(t)\}$
  sample K batch $\{\hat{x}_k, d\hat{x}_k, u_k\}_{k=1}^K$
  $\theta^{i+1} \leftarrow \theta^i + \sum_k \nabla_\theta \log \mathcal{L}$ given batch
  update $p(s)$ from $\sigma_{\theta^{i+1}}$
  $i \leftarrow i+1$

REFERENCES

[1] H. Axelsson, Y. Wardi, M. Egerstedt, and E. Verriest, "Gradient descent approach to optimal mode scheduling in hybrid dynamical systems," *Journal of Optimization Theory and Applications*, vol. 136, no. 2, pp. 167–186, 2008.

[2] L. M. Miller, Y. Silverman, M. A. MacIver, and T. D. Murphey, "Ergodic exploration of distributed information," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 36–52, 2016.

[3] A. Mavrommati, E. Tzorakoleftherakis, I. Abraham, and T. D. Murphey, "Real-time area coverage and target localization using receding-horizon ergodic exploration," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 62–80, 2018.

[4] E. Ayvali, H. Salman, and H. Choset, "Ergodic coverage in constrained environments using stochastic trajectory optimization," in *International Conference on Intelligent Robots and Systems*, 2017, pp. 5204–5210.

[5] I. Abraham, A. Prabhakar, M. J. Hartmann, and T. D. Murphey, "Ergodic exploration using binary sensing for nonparametric shape estimation," *IEEE robotics and automation letters*, vol. 2, no. 2, pp. 827–834, 2017.

[6] I. Abraham, A. Mavrommati, and T. D. Murphey, "Data-driven measurement models for active localization in sparse environments," in *Robotics: Science and Systems*, 2018.

[7] I. Abraham and T. D. Murphey, "Decentralized ergodic control: distribution-driven sensing and exploration for multiagent systems," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2987–2994, 2018.

[8] A. Polyakov and L. Fridman, "Stability notions and Lyapunov functions for sliding mode control systems," *Journal of the Franklin Institute*, vol. 351, no. 4, pp. 1831–1865, 2014.

[9] P. Kormushev, S. Calinon, and D. G. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning," in *International Conference on Intelligent Robots and Systems*, 2010, pp. 3232–3237.

[10] R. F. Reinhart, "Autonomous exploration of motor skills by skill babbling," *Autonomous Robots*, vol. 41, no. 7, pp. 1521–1537, 2017.

[11] C. D. McKinnon and A. P. Schoellig, "Learning multimodal models for robot dynamics online with a mixture of Gaussian process experts," in *International Conference on Robotics and Automation*, 2017, pp. 322–328.

[12] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," in *Proceedings of Robotics: Science and Systems*, 2018.

[13] A. Marco, F. Berkenkamp, P. Hennig, A. P. Schoellig, A. Krause, S. Schaal, and S. Trimpe, "Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with Bayesian optimization," in *International Conference on Robotics and Automation*, 2017, pp. 1557–1563.

Algorithm 5 KL-E^3 enhanced DDPG

init: see Alg. 1, [19],
while task not done **do**
  set $x(t_i) = \hat{x}(t_i)$
  ▷ simulation loop (see Alg. 1 lines 4-18)
  get $\delta\mu(t)$ from simulation
  ▷ apply to real robot
  chose $\tau \in [t_i, t_i + t_H]$ and $\lambda \leq t_H$
  for $t \in [t_i, t_{i+1}]$ **do**
    if $t \in [\tau, \tau + \lambda]$ **then**
      apply $\mu_\star(t) = \delta\mu_\star(t) + \mu(\hat{x}(t))$
    else
      apply $\mu(x(t))$
    if time to sample **then**
      measure state $\hat{x}(t)$, next state $\hat{x}'(t)$, applied control $u(t)$, reward $r(t)$
      append $\mathcal{D} \leftarrow \{\hat{x}(t), \hat{x}'(t), u(t), r(t)\}$
  if time to update and buffer is large enough **then**
    update Q, μ from [19]
    update $p(s)$ using Q (41)
  $i \leftarrow i + 1$

[14] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Advances in Neural Information Processing Systems*, 2017, pp. 908–918.

[15] T.-C. Lin and Y.-C. Liu, "Direct learning coverage control based on expectation maximization in wireless sensor and robot network," in *Conference on Control Technology and Applications*, 2017, pp. 1784–1790.

[16] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *International Conference on Intelligent Robots and Systems*, vol. 1, 2002, pp. 540–545.

[17] R. Arnold and A. Wellerding, "On the sobolev distance of convex bodies," *aequationes mathematicae*, vol. 44, no. 1, pp. 72–83, 1992.

[18] D. Precup, R. S. Sutton, and S. Dasgupta, "Off-policy temporal-difference learning with function approximation," in *ICML*, 2001, pp. 417–424.

[19] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[20] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[21] P. I. Frazier, "A tutorial on Bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[22] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[23] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "Bayesian optimization for learning gaits under uncertainty," *Annals of Mathematics and Artificial Intelligence*, vol. 76, no. 1-2, pp. 5–23, 2016.

[24] M. Schwager, P. Dames, D. Rus, and V. Kumar, "A multi-robot control policy for information gathering in the presence of unknown hazards," in *Robotics research*, 2017, pp. 455–472.

[25] D. Nguyen-Tuong and J. Peters, "Incremental online sparsification for model learning in real-time robot control," *Neurocomputing*, vol. 74, no. 11, pp. 1859–1867, 2011.

[26] D. Ucinski, *Optimal measurement methods for distributed parameter system identification*. CRC Press, 2004.

[27] G. Mathew and I. Mezić, "Metrics for ergodicity and design of ergodic dynamics for multi-agent systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 4-5, pp. 432–442, 2011.

[28] L. M. Miller and T. D. Murphey, "Trajectory optimization for continuous ergodic exploration," in *2013 American Control Conference*, 2013, pp. 4196–4201.

[29] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[30] S. Kullback, *Information theory and statistics*. Courier Corporation, 1997.

[31] E. D. Sontag, "Control-Lyapunov functions," in *Open problems in mathematical systems and control theory*, 1999, pp. 211–216.

[32] S. M. Khansari-Zadeh and A. Billard, "Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.

[33] Z. Artstein, "Stabilization with relaxed controls," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 7, no. 11, pp. 1163–1173, 1983.

[34] I. Abraham, A. Prabhakar, and T. D. Murphey, "Active area coverage from equilibrium," in *Workshop on Algorithmic Foundations of Robotics*, 2019.

[35] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[36] J. J. Moré and D. J. Thuente, "Line search algorithms with guaranteed sufficient decrease," *ACM Transactions on Mathematical Software (TOMS)*, vol. 20, no. 3, pp. 286–307, 1994.

[37] W. Zhong and H. Rock, "Energy and passivity based control of the double inverted pendulum on a cart," in *IEEE International Conference on Control Applications*, 2001, pp. 896–901.

[38] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[39] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[40] I. Abraham and T. D. Murphey, "Active learning of dynamics for data-driven control using koopman operators," *arXiv preprint arXiv:1906.05194*, 2019.

[41] T. Fan and T. Murphey, "Online feedback control for input-saturated robotic systems on lie groups," *arXiv preprint arXiv:1709.00376*, 2017.

[42] Y. Gal, R. McAllister, and C. E. Rasmussen, "Improving pilco with Bayesian neural network dynamics models," in *Data-Efficient Machine Learning workshop, ICML*, vol. 4, 2016.

[43] A. D. Wilson, J. A. Schultz, A. R. Ansari, and T. D. Murphey, "Dynamic task execution using active parameter identification with the baxter research robot," *Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 391–397, 2017.

[44] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the brownian motion," *Physical review*, vol. 36, no. 5, p. 823, 1930.

[45] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1433–1440.

[46] E. Bıyık and D. Sadigh, "Batch active preference-based learning of reward functions," *arXiv preprint arXiv:1810.04303*, 2018.

[47] D. S. Brown, Y. Cui, and S. Niekum, "Risk-aware active inverse reinforcement learning," *arXiv preprint arXiv:1901.02161*, 2019.