Symplectic Integration for Optimal Ergodic Control

Ahalya Prabhakar, Kathrin Flaßkamp, and Todd D. Murphey

***Abstract*— Autonomous active exploration requires search algorithms that can effectively balance the need for workspace coverage with energetic costs. We present a strategy for planning optimal search trajectories with respect to the distribution of expected information over a workspace. We formulate an iterative optimal control algorithm for general nonlinear dynamics, where the metric for information gain is the difference between the spatial distribution and the statistical representation of the time-averaged trajectory, i.e. ergodicity. Previous work has designed a continuous-time trajectory optimization algorithm. In this paper, we derive two discrete-time iterative trajectory optimization approaches, one based on standard first-order discretization and the other using symplectic integration. The discrete-time methods based on first-order discretization techniques are both faster than the continuous-time method in the studied examples. Moreover, we show that even for a simple system, the choice of discretization has a dramatic impact on the resulting control and state trajectories. While the standard discretization method turns unstable, the symplectic method, which is structure-preserving, achieves lower values for the objective.**

I. INTRODUCTION

Autonomous active exploration relies on the development of efficient search algorithms. In exploration, there is a constant tradeoff between coverage of the search area and energetic cost. In an ideal setting, where there are no limits on energy or time, the robot would explore the entire search area, taking a large number of high quality measurements everywhere in the domain. There are many methods that use this goal to derive uniform coverage strategies [6]. However, in applications, there is a cost for time and energy expenditure. Uniform coverage strategies may produce trajectories that are energetically expensive or dynamically infeasible for the sensor to perform. Also, coverage does not take into account the potential need to revisit an area or to take a higher density of measurements in a particular region. These realistic limits drive the need for search strategies that efficiently explore a region taking into account the information density of the space.

Previous work has developed a method of generating continuous-time, dynamically constrained search strategies. We use projection-based trajectory methods to derive trajectories that distribute the measurements proportional to the spatial distribution of the expected information method [18], [19]. However, this continuous method nominally requires the exact solution of a differential equation, which makes the numerical implementation of the algorithm rely on numerical approximations. In this work, we develop a method of generating dynamically constrained search strategies using an analogous discrete method. We assume one knows the distribution of information in a given workspace. We choose to use an ergodic metric representing the difference between the time-averaged behaviour and the spatial probability density function of the information density, as defined by Mathew and Mezic [16], described in Section III.

In application scenarios such as underwater exploration by autonomous robots (see e.g. [21]), ergodic control problems have to be solved iteratively on a receding horizon in order to take into account the updated information density. Therefore, efficient algorithms are needed that provide fast system simulations even for complex dynamics. For this reason, first-order methods, typically the explicit Euler scheme, are chosen to replace time-consuming simulations with higher-order numerical integration methods. However, the explicit Euler scheme is known to provide bad state reconstructions and turns unstable when used with large step-sizes. Thus, in this contribution, we also study the combination of the discrete-time trajectory optimization with a variational integrator of first order, the symplectic Euler integrator. Unlike standard integration schemes, variational integrators are derived from mechanical variational principles, and therefore preserve system structures such as symplecticity, symmetry/momentum, constraints, statistical properties, and have a good long-term energy behavior [8], [15]. Our simulation results show that the symplectic Euler scheme leads to an ergodic control method which is computationally efficient *and* provides reliable trajectories.

We begin with a discussion of related work in Section II. We then describe ergodicity as a metric for representing information density in Section III. Section IV includes an overview of ergodic trajectory optimization in continuous time, as presented in previous work, and in discrete time. It also includes a brief overview of integration schemes used in this work (explicit Euler and symplectic Euler). We present numerical examples in Section V, including comparisons between continuous-time and discrete-time ergodic trajectory optimization using different integration schemes.

II. RELATED WORK

The quality of measurements depends on the sensors, and may relate to their distance, orientation, or motion [4], [22]. Informative planning methods have employed different metrics to encode information density and quality of measurements in a search space, and algorithms seek to optimize their control strategies with respect to these metrics. Some have used Shannon entropy [2], [23], mutual information

A. Prabhakar, K. Flaßkamp, and T.D. Murphey are with the Department of Mechanical Engineering, Northwestern University, Evanston, IL60208, USA AhalyaPrabhakar2013@u.northwestern.edu, {kathrin.flasskamp,t-murphey}@northwestern.edu

[14], or Fisher Information [17] to represent the information density.

Methods of exploring search spaces typically involve balancing the exploration of the space with the feasibility and cost of the control effort. Some methods of determining search strategies highly prioritize exploration (i.e., lawnmower search strategies [6]), typically incurring high control costs or producing trajectories that are infeasible when considering the dynamics of the system. Instead, many methods use different techniques to balance the control cost with the search trajectory.

Most methods require an expectation of the belief and control, which is computationally expensive, requiring different strategies to make them computationally tractable. Some methods require the decomposition or discretization of the workspace to represent the workspace in a tractable setting [6]. Other methods build the control trajectory iteratively by determining the next, single optimal control step locally, rather than optimizing the full control trajectory [7]. These methods, however, are particularly vulnerable to local information maxima or regions of high uncertainty. To mitigate the effect of local extrema, many methods use control strategies over longer time horizons- nonmyopic methods. Some nonmyopic algorithms use dynamic programming [5], which is computationally expensive, or heuristics to approximate the dynamic programming results, which are not necessarily accurate [5], [13]. Some methods optimize the expected measurement utility using a set of predefined candidate control actions or a search graph [10]. However, these techniques that use computationally tractable methods may produce trajectories that, while feasible, have a high energetic cost to execute.

In addition, it is often difficult to incorporate the dynamics of the mobile sensor itself. Some methods ensure feasibility by combining the planning methods with feedback controllers [14] or using a search graph with a predefined set of feasible control actions [10]. While many of the methods described above optimize with respect to the information density and account for the feasibility of the various control actions, they do not necessarily directly optimize with respect to the control costs or dynamics, producing trajectories that may be feasible and explorative, but nevertheless costly to execute.

The algorithm presented in this paper uses first-order integration methods in order to remain computationally tractable, particularly for high-dimensional or large-scale systems. The ergodic metric addresses the need for informative planning, producing trajectories that not only cover regions of interest efficiently, but spend a longer amount of time in those regions in order to obtain sufficient measurements to explore these information-dense regions. Furthermore, the algorithm optimizes with respect to the dynamics of the sensor, including them as a constraint to the optimization problem. This allows the algorithm to generate trajectories that are not only feasible, but are dynamically efficient for the sensor to execute, reducing the cost of the control signal.

III. Ergodicity

As mentioned in Section I, a trajectory is *ergodic* with respect to the distribution if the time spent in a region is proportional to the information density of the region; ergodicity encodes the idea that trajectories should spend more time in highly informative regions. The spatial distribution is defined by a probability density function (PDF) and denoted by $\phi(x)$.

We define the metric to be minimized as the *distance from ergodicity* ε of the trajectory $(x(t))$. The ergodicity of a trajectory can be quantified as the sum of the weighted squared distance between the Fourier coefficients of the spatial distribution ϕ_k and the distribution representing the trajectory c_k, defined below:

$$\varepsilon = \sum_{k_1=0}^{K} \cdots \sum_{k_n=0}^{K} \Lambda_k |c_k - \phi_k|^2, \tag{1}$$

where $K+1$ is the number of coefficients (nominally infinite, but $K < \infty$ in computation) along each of the n dimensions and k labels the set of all combinations. The coefficients Λ_k place a larger weight on lower frequency information and are defined as $\Lambda_k = \frac{1}{(1+||k||^2)^s}$, where $s = \frac{n+1}{2}$ [16].

To compute the Fourier basis functions, we use

$$F_k(x) = \frac{1}{h_k} \prod_{i=1}^{n} \cos\left(\frac{k_i \pi}{L_i} x_i\right), \tag{2}$$

where h_k is a normalizing factor, as defined in [16].

The Fourier coefficients ϕ_k of the spatial distribution $\phi(\cdot)$ are determined using an inner product

$$\phi_k = \int_X \phi(x) F_k(x) dx,$$

and the Fourier coefficients of the basis function along the time-averaged trajectory $x(\cdot)$ are computed as

$$c_k = \frac{1}{T} \int_0^T F_k(x(t)) dt,$$

where T is the final time [16]. One thing to note is that the Fourier basis functions are periodic in nature, causing the reconstruction of the PDF representing the spatial distribution using these basis functions to be periodic as well. This will be important in Section V-B.

IV. Trajectory Optimization

Using a projection-based method [9], we can define a local quadratic model of the ergodic objective function and calculate the steepest descent direction to use in iterative first-order optimization methods. The analogous discrete projection-based optimization method is derived from [11].

The sections below describe the system dynamics, ergodic objective function, and the trajectory optimization for both the continuous and discrete cases.

A. Continuous-time Projection-based Trajectory Optimization

The dynamics of a general, nonlinear dynamic mobile sensor can be modeled as $\dot{x}(t) = f(x(t), u(t))$, where $x \in \mathbb{R}^S$ denotes the state and $u \in \mathbb{R}^M$ denotes the control.

The objective function $J(\cdot)$ is comprised of the ergodic metric as defined in Eq. (1) and the integrated magnitude of the control effort, and takes as an argument the dynamically unconstrained curve $\xi = (\alpha, \mu)$,

$$J(\xi(\cdot)) = q \cdot \sum_{k=0}^{K} \Lambda_k \left(\frac{1}{T} \int_0^T F_k(\alpha(t))\, dt - \phi_k \right)^2 + \int_0^T \frac{1}{2} \mu(t)^T R(t) \mu(t)\, dt, \quad (3)$$

where $q \in \mathbb{R}$ represents the weight of the ergodic metric [18], [19]. Let $\mathcal{T}$ denote the trajectory manifold of curves $\xi = (x, u)$ which satisfy $\dot{x}(t) = f(x, u)$ and $x(0) = x_0$. The goal is to find a feasible trajectory that minimizes the objective function, i.e.

$$\arg \min_{\xi(\cdot) \in \mathcal{T}} J(\xi(\cdot)). \quad (4)$$

The optimization in Eq. (4) can be reformulated as an unconstrained trajectory optimization problem using the projection operator from [9]. We define the projection operator as

$$P(\xi(\cdot)) : \begin{cases} u(t) = \mu(t) + K(t)(\alpha(t) - x(t)) \\ \dot{x}(t) = f(x(t), u(t)),\ x(0) = x_0. \end{cases} \quad (5)$$

The optimal feedback gain $K(t)$ is computed by solving a linear quadratic regulator problem. Using the projection operator, the optimization problem can be reformulated with the goal to minimize $J(P(\xi(\cdot)))$. This reformulation has the benefit of removing the nonlinear constraints from the dynamics during the descent direction.

The descent direction $\zeta_i(\cdot)$ must be calculated at each iteration i in order to use first-order optimization methods. To determine the descent direction, we minimize the quadratic model of the form

$$\zeta_i^*(\cdot) = \arg \min_{\zeta_i} \int_0^T a^T(\tau) z_i + b^T(\tau) v_i + \tfrac{1}{2} z_i(\tau)^T Q_n(\tau) z_i(\tau) + \tfrac{1}{2} v_i(\tau)^T R_n(\tau) v_i(\tau) d\tau, \quad (6)$$

subject to the differential equation $\dot{z} = Az + Bv$, where $A = D_1 f(x(t), u(t))$ and $B = D_2 f(x(t), u(t))$, since the descent direction $\zeta_i(\cdot)$ is constrained to lie in the tangent space of the trajectory manifold. a is the derivative of the objective function (Eq. (3)) with respect to $x(\cdot)$ and b is the derivative of the objective function (Eq. (3)) with respect to $u(\cdot)$. Q and R are both symmetric matrices that represent the weight on the state and control, respectively. The basic algorithm is outlined in previous work [19], and is analogous to the discrete version described in Algorithm 1, Section IV-B.

B. Discrete-time Projection-based Trajectory Optimization

For an implementation of the ergodic trajectory optimization, numerical integration methods are required to solve the LQR/LQ problems. Alternatively, one starts with a discretization of the system dynamics and the cost function, such that discrete-time LQR/LQ theory can be applied (cf. [1], [11]). We follow the latter approach because we want to explicitly choose the integration method that leads to the discrete-time model. As it turns out in the simulated examples (Section V), this choice is crucial for the algorithm's performance.

In the discrete-time case, the mobile sensor is modeled as

$$x_{n+1} = g(x_n, u_n, t_n), \quad (7)$$

for $n = 0, 1, \ldots, N-1$ on a discretized time grid $\Delta = \{t_0, t_1, \ldots, t_N\}$ with $t_0 = 0$, $t_N = T$ and given initial point $x_0 = x^0$. The discrete-time model $g(\cdot, \cdot, \cdot)$ can be obtained from applying an integration scheme to the continuous time dynamics as shown in Section IV-C. For the discrete-time trajectory optimization method, linearizations of the discrete dynamics are required, which we represent in discrete state-space form

$$x_{n+1} = A_n x_n + B_n u_n, \quad \text{for } n = 0, 1, \ldots, N-1. \quad (8)$$

In the discrete case, the time integral of the continuous objective function, (3), i.e. the c_k term and the control effort, is approximated by the left Riemann sum,

$$J = q \cdot \sum_{k=0}^{K} \Lambda_k \left(\frac{1}{T} \sum_{n=1}^{N-1} h \cdot F_k(x_n) - \phi_k \right)^2 + \sum_{n=1}^{N-1} \frac{h}{2} u_n^T R_n u_n, \quad (9)$$

where N is the total number of discrete points, $h = t_{n+1} - t_n$ for $n = 0, 1, \ldots, N-1$ is the step size (assumed to be constant for ease of notation), and the Fourier basis functions F_k (Eq. (2)) are now evaluated at the discrete states x_n.

The discrete time projection-based method is similar to the continuous time and is described in Algorithm 2. However, rather than yielding a continuous trajectory $\xi = (x, u)$, the output from this method is of the form $\xi_d = (x_d, u_d)$, where $x_d = \{x_0, x_1, \ldots, x_N\}$ and $u_d = \{u_0, u_1, \ldots, u_N\}$ are discrete state and control trajectories. These optimal

Algorithm 1 First-Order Descent for Discrete Ergodic Trajectory Optimization

Calculate ϕ_k for $k = 0, 1, \ldots, K$
Initialize $\xi_{d,0} \in \mathcal{T}_d$, tolerance ϵ
while $DJ(\xi_{d,i}) \circ \zeta_{d,i} > \epsilon$ **do**
  Calculate optimal gains $K_0, \ldots, K_N$ for the projection operator (LQR problem):
    Solve discrete Riccati Equation for P_n, $n = 0, 1, \ldots, N$
  Calculate descent direction (LQ Problem):
    $\zeta_i = \arg \min_{\zeta_i \in T_{\xi_i} \mathcal{T}} DJ(\xi_i) \circ \zeta_i + \frac{1}{2} \langle \zeta_i, \zeta_i \rangle$
    Solve discrete Riccati Equations for P_n, r_n $(n = 0, 1, \ldots, N)$
    Use P_n, r_n to find descent directions
  Calculate γ_i using Armijo line search [3]
  Project the update:
    $\xi_{d,i+1} = \mathcal{P}(\xi_{d,i} + \gamma_i \zeta_{d,i})$
  $i = i + 1$
end while

trajectories can be calculated by solving discrete Riccati equations to determine the optimal gains K_n and the descent direction ζ_d in every iteration i.

The projection operator used in the discrete case, $P : (\alpha_d, \mu_d) \to (x_d, u_d)$ (for $\alpha_d = \{\alpha_0, \alpha_1, \ldots, \alpha_N\}$, $\mu_d = \{\mu_0, \mu_1, \ldots, \mu_N\}$) is analagous to the continuous-time version,

$$P(\xi_d) : \begin{cases} u_n = \mu_n + K_n(\alpha_n - x_n) \\ x_{n+1} = g(x_n, u_n, t_n),\ x_0 = x^0, \\ \text{for } n = 0, 1, \ldots, N-1, \end{cases} \tag{10}$$

where $g(\cdot,\cdot,\cdot)$ is the discrete model depending on the chosen integration scheme and x^0 the known initial point. To calculate the optimal projection gains $K_0, K_1, \ldots, K_{N-1}$, we solve a discrete linear quadratic regulator problem with the objective function

$$J(x_d, u_d) = \sum_{n=0}^{N-1} [x_n^T Q_n x_n + u_n^T R_n u_n] + x_N^T Q_N x_N, \tag{11}$$

where R_n and Q_n are symmetric, positive semidefinite for all $n = 0, 1, \ldots, N-1$. The optimal gains K_n are determined using the discrete backwards Riccati-like equation:

$$\begin{aligned} &K_n = \Gamma_n^{-1} B_n^T P_{n+1} A_n, \text{ for } n = 0, 1, \ldots, N-1 \\ &\text{with } \Gamma_n = R_n + B_n^T P_{n+1} B_n, \\ &\text{and } P_N = Q_N, \\ &P_n = Q_n + A_n^T P_{n+1} A_n - K_n^T \Gamma_n K_n. \end{aligned} \tag{12}$$

The descent direction can be found by solving a discrete-time linear quadratic (LQ) optimization problem, where the linear quadratic cost is

$$J(\zeta_d) = \sum_{n=0}^{N-1} [a_n^T z_n + b_n^T v_n + z_n^T Q_n z_n + v_n^T R_n v_n] + z_N^T P_N z_N, \tag{13}$$

with discrete curve $\zeta_d = (z_d, v_d)$ consisting of the discrete states $z_d = \{z_0, z_1, \ldots, z_N\}$ and controls $v_d = \{v_0, v_1, \ldots, v_N\}$. As in the continuous-time case, a_n represent the derivatives of the objective function (Eq. (9)) with respect to the state in discrete time and b_n represent the derivative of the objective function (Eq. (9)) with respect to the control in discrete time. Because the descent direction is constrained to the tangent space of the (discrete-time) trajectory manifold, it must satisfy the constraints from the discrete model

$$z_{n+1} = A_n z_n + B_n v_n, \text{ for } k = 0, 1, \ldots, N-1, \tag{14}$$

where A_n and B_n represent the explicit state-space form for the discrete dynamics linearized at the current discrete trajectory (x_d, u_d).

To solve the LQ problem of the descent direction, we use pairs of discrete, Riccati-like equations for matrices P_n and vectors r_n (cf. [1]). We solve for P_n, K_n, and Γ_n as defined in Eq. (12), and for r_n using

$$\begin{aligned} r_N &= a_N \\ r_n &= [A_n^T - K_n^T B_n^T] r_{n+1} + a_n - K_n^T b_n. \end{aligned} \tag{15}$$

Once P_n and r_n are solved for all $n = 0, 1, \ldots, N$, we obtain the descent direction $\zeta_d = (z_d, v_d)$ from

$$\begin{aligned} v_n &= b_n + B_n^T P_{n+1} z_n + B_n^T r_{n+1} \\ z_{n+1} &= A_n z_n + B_n v_n. \end{aligned}$$

Up until now, we have simply showed how one can adapt continuous-time ergodic trajectory optimization to a discrete-time setting. What is important to note is that the choice of integrator has a profound impact on the quality of solution, even if one is choosing between two first-order methods (Section IV-C), as demonstrated in Section V-B.

C. Integration Schemes

As mentioned in Section IV-B, the discrete-time model (7) can be obtained by discretizing the continuous-time model with an numerical integration scheme. In this work, we evaluate the discrete trajectory optimization method using two types of first-order integration schemes: the forward explicit Euler and the symplectic Euler, which is a variational integration method. For ease of notation, we assume an equidistant time grid Δ with constant step-size h.

The forward explicit Euler integration scheme approximates the continuous time system, $\dot{x} = f(x, u)$, $x(0) = x^0$, by

$$x_{k+1} = x_k + h \cdot f(x_k, u_k),$$

with $x_0 = x^0$ and u_k being the discrete input to the system at $t_k = hk$. We also linearize the scheme to the form $x_{k+1} = A_k x_k + B_k u_k$, where $A_k = I + hD_1 f(x_k, u_k)$ and $B_k = hD_2 f(x_k, u_k)$.

Variational integrators (VI) are tailored to the integration of mechanical systems, as they preserve properties of the original continuous-time dynamics, such as symplecticity or symmetries (i.e. momentum) [8], [15], [20] independent of the chosen step size. Therefore, VIs can be beneficially used in real-time control methods [12]. Moreover, VIs are guaranteed to have a good energy behavior even for long-term simulations, i.e. there is no artifical increase or decrease in the system's energy as observed in general Runge-Kutta schemes [8].

The symplectic Euler method can be defined by considering the Hamiltonian formulation of the mechanical system in configuration-momentum (q, p) coordinates. These are

$$\dot{q} = \frac{\partial H}{\partial p}, \qquad \dot{p} = -\frac{\partial H}{\partial q} + f_H(q, p, u),$$

where q represents the configurations, p represents the momenta, u represents the control, H represents the Hamiltonian, and f_H represents the state- and control-dependent forcing. The *symplectic Euler* integration scheme (cf. [8]) is a first-order symplectic/variational integrator defined as

$$\begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} q_n \\ p_n \end{pmatrix} + h \cdot \begin{pmatrix} \frac{\partial}{\partial p} H(q_n, p_{n+1}) \\ -\frac{\partial}{\partial q} H(q_n, p_{n+1}) + f_H(q_n, p_{n+1}, u_n) \end{pmatrix}.$$

Note that this is an implicit update equation in the momentum coordinates p_{n+1}. For computing the projection and the descent direction in the discrete-time optimization method, the discrete-time system has to be linearized and tranformed into explicit state space form (cf. (8)),

$$\begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} A_{d11}^n & A_{d12}^n \\ A_{d21}^n & A_{d22}^n \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix} + \begin{pmatrix} B_{d1} \\ B_{d2} \end{pmatrix} u_n. \quad (16)$$

The block matrices of A_d and B_d for the linearization are given by

$$\begin{aligned} A_{d11}^n &= I + hA_{11}^n + h^2 A_{12}^n (I - hA_{22}^n)^{-1} A_{21}^n \\ A_{d12}^n &= hA_{12}^n (I - hA_{22}^n)^{-1} \\ A_{d21}^n &= h(I - hA_{22}^n)^{-1} A_{21}^n \\ A_{d22}^n &= (I - hA_{22}^n)^{-1} \\ B_{d1}^n &= h^2 A_{12}^n (I - hA_{22}^n)^{-1} B_2^n + hB_1^n \\ B_{d2}^n &= h(I - hA_{22}^n)^{-1} B_2^n. \end{aligned}$$

V. Simulated Examples

We present two simulated examples of discrete ergodic trajectory optimization. We evaluate the algorithm for two second order, linear mechanical systems with two degrees of freedom. We apply the trajectory optimization method using both the forward explicit Euler and symplectic Euler integrator schemes for a sensor exploring a PDF in a two dimensional space and compare them to the result from continuous-time ergodic trajectory optimization[1].

A. Double Integrator in $\mathbb{R}^2$

The state for this model is $X = [x, y, \dot{x}, \dot{y}]^T$, where x and y are Cartesian coordinates and $\dot{x}$ and $\dot{y}$ are the velocities in the x- and y-directions. We model a double integrator system with continuous-time dynamics[2]

$$f(X) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_x \\ u_y \end{bmatrix}. \quad (17)$$

For this example, we use a 30 second time horizon, a step size $h = .6s$ and an initial trajectory (x_d, u_d) obtained from forward integration with constant $u_n = (.0001, .0001)$, starting at $x^0 = [0.5, 0.01]^T$. We consider $K = 5$ Fourier coefficients in $\mathbb{R}^2$. The problem puts a larger weight on the ergodic metric with $q = 1$ and $R = .01 \cdot I_2$, where I_2 is the 2×2 identity matrix, and Algorithm 1 runs for 100 iterations.

Figure 1 shows the optimally ergodic trajectories for the dynamics in Eq. (17) with respect to the multimodal Gaussian PDF. We display both the point of the discrete trajectory, as well as the interpolation of this trajectory in continuous time. The continuous-time optimal trajectory is shown in green, and the explicit Euler- and symplectic Euler-discretized optimal trajectories are shown in red and blue respectively. Both discrete methods produce trajectories that are similar to the continuous-time solution. We can also see quantitively in Table I that the costs of the trajectories are similar. [3]

	Continuous	Explicit Euler	Variational
Cost	0.003496	0.004734	0.003944
Control Cost	0.0000355	0.0000913	0.0000775
Ergodic Metric	0.00346	0.00464	0.003944
Run Time (sec)	257.584	52.223	52.821

TABLE I

Comparison of metrics for discrete and continuous trajectory optimization schemes for example 1.

Because of the simplistic nature of the mechanical system, the implicit nature of the symplectic Euler integration scheme –compared to the explicit Euler solution – has no major effect on the results, though the trajectories have a marginally lower cost, as shown in Table I. Moreover, both integration schemes produce similar control strategies, highlighting the fact that, for this simple system, both integration schemes produce nearly optimal trajectories.

However, the run times in the both discrete cases are significantly faster, by roughly a factor of 5 in Mathematica, than the continuous time trajectory optimization.

B. Undamped Oscillator in $\mathbb{R}^2$

The previous example suggests that the choice of integrator has little effect on control. Now, we demonstrate the effect of the different integration schemes using a double integrator system with an undamped oscillator in both the x- and y-dimensions. As in the first example, the state of this system is $X = [x, y, \dot{x}, \dot{y}]^T$, where x and y are cartesian coordinates and $\dot{x}$ and $\dot{y}$ are the forward velocities in the x- and y-directions. We model this system in continuous time state-space form as:

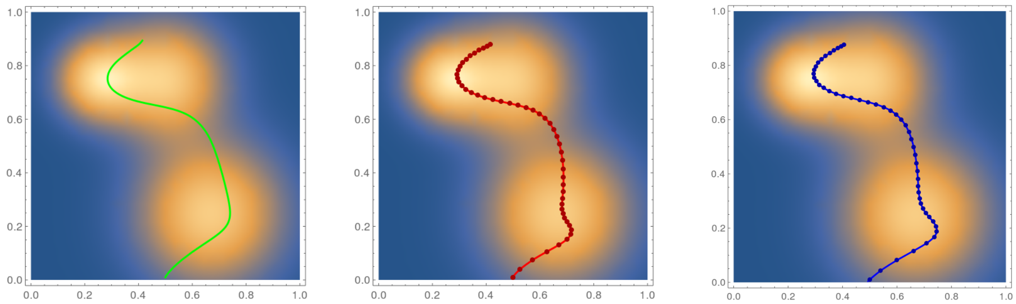
$$f(X) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -k_1 & 0 & 0 & 0 \\ 0 & -k_2 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_x \\ u_y \end{bmatrix}, \quad (18)$$

where $k_1 = 1$ and $k_2 = 1$ represent the spring constants in the x- and y-dimensions respectively. We use a 30 second time horizon, $h = .6$, and initial control trajectory of constant $u = (.0001, .0001)$, starting at $x^0 = [0, -0.49]^T$. $K = 5$ Fourier coefficients are again used. The weights in the objective function are again $q = 1$ and $R = .01 \cdot I_2$. We also tested this example using 10 alternative starting points on a grid in the workspace, with similar results for all runs.

[1]The continuous method uses a high-order, adaptive, black box integration routine provided by Mathematica.

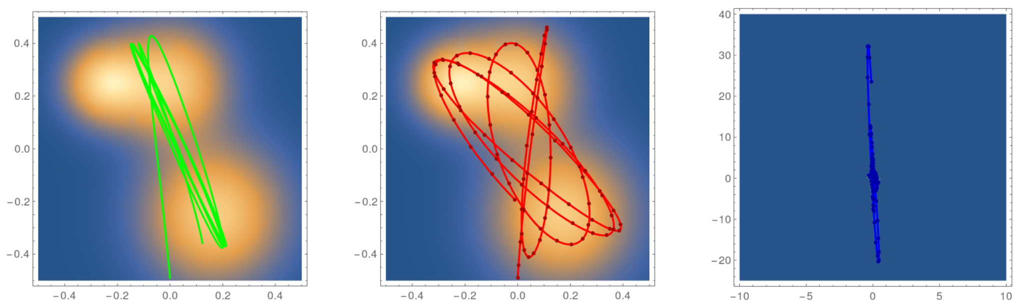
[2]For the symplectic Euler integrator, Hamilton's equations are needed. However, because this system treats the sensor as a point mass and it moves with linear translational motion in two degrees of freedom, one can trivially transform between momenta and velocities. By choosing the mass equal to 1, we can use the vector field as defined. Otherwise, transforming the system to momentum coordinates requires a scalar multiplication only, due to the simplicity of the system's Legendre transform.

[3]We calculate these costs using the interpolations of the output optimal trajectories and control signals over the 30 second time horizon. The costs are calculated using the continuous-time cost function metrics with the interpolations as inputs.



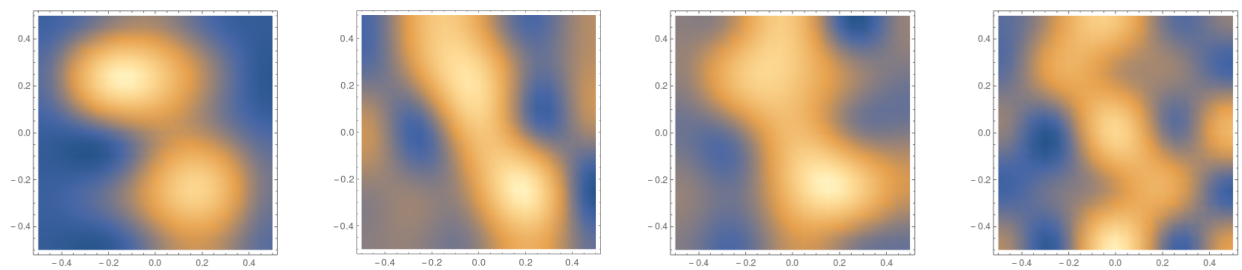
(a) Continuous (b) Symplectic Euler (c) Explicit Euler

Fig. 1. Optimal Trajectories for continuous-time and discrete-time methods for multimodal PDF and dynamics of Eq. (17).



(a) Continuous (b) Symplectic Euler (c) Explicit Euler

Fig. 2. Optimal trajectories from continuous-time and discrete-time ergodic trajectory optimization for multimodal PDF and dynamics of Eq. (18).



(a) PDF from spatial distribution (b) PDF from continuous-time optimal trajectory (c) PDF from symplectic Euler optimal trajectory (d) PDF from explicit Euler optimal trajectory

Fig. 3. PDF reconstruction from discrete-time methods for multimodal PDF and dynamics of Eq. (17) and Eq. (18). Fig. 3(a) displays the PDF based on the Fourier coefficients ϕ_k from the spatial distribution. Fig. 3(b), 3(c) and 3(d) display the PDFs based on the Fourier coefficients c_k from the time-averaged trajectories.

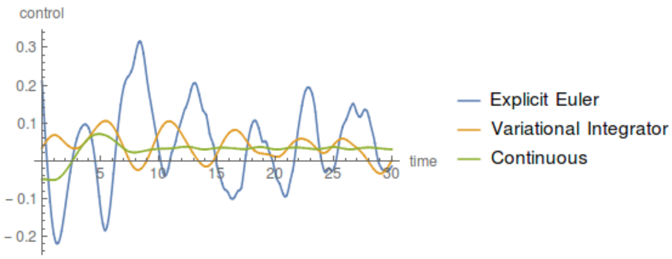
	Continuous	Explicit Euler	Variational
Cost	0.007416	0.004790	0.000907
Control Cost	0.000264	0.002383	0.000443
Ergodic Metric	0.00702	0.00717	0.000463
Run Time (sec)	440.742	139.410	140.244

TABLE II

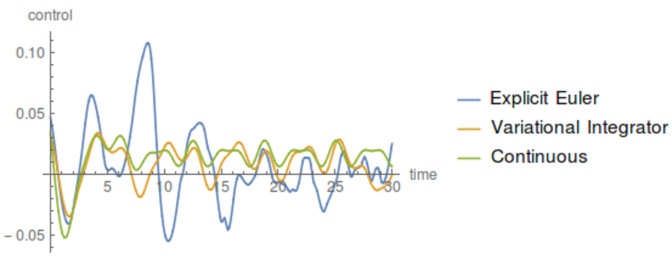
COMPARISON OF METRICS FOR DISCRETE AND CONTINUOUS TRAJECTORY OPTIMIZATION SCHEMES FOR EXAMPLE 2.

Figure 2 demonstrates the profound impact the choice of numerical integration scheme can have on ergodic trajectory synthesis. While symplectic Euler provides an optimized trajectory close to, and somewhat better than, continuous-time trajectory optimization, the trajectory optimization based on explicit Euler diverges quickly from the origin. Moreover, the optimal control signals (Fig. 4) are substantially larger (Table II). Nevertheless, the ergodic metric and total cost (Table II) are comparable to both continuous time and symplectic Euler. The reason for this inconsistency is that the ergodic metric in Eq. (1) is periodic in $\mathbb{R}^2$ because it is computed using Fourier transforms. Hence, the energetic instability of explicit Euler actually creates a situation where the optimal solution is to use small amounts of control authority to control how divergence passes through the periodic distribution, rather than use a lot of control authority to drive the system back to a neighborhood of the origin. Figure 3 illustrates that trajectory optimization based on explicit Euler is doing exactly that–despite divergence from the origin, the transform of explicit Euler in Fig. 3(d) is very similar to the desired reconstructed PDF in Fig. 3(a).

This example highlights the importance of the choice of



(a) Control in x-dimension



(b) Control in y-dimension

Fig. 4. Comparison of control signals in each dimension for continuous and discrete trajectory optimization methods for Eq. (18).

discretization. The symplectic Euler as a variational integrator does not artificially add energy to the system, as the explicit Euler integration does for this large time step of $h = .6$. Moreover, the optimal trajectory produced is stable and not artificially ergodic, as is the case for the explicit Euler example. The periodic nature of the ergodic metric requires careful choice in integration schemes in order to ensure that the trajectories produced are not unstable. Though both discrete methods have a much lower run time than the continuous-time method, only the symplectic Euler solution is plausible for implementation.

VI. CONCLUSIONS

We derive a strategy for planning exploratory path to sample a workspace efficiently with respect to its spatial sensory information density over the space. Previous work with ergodic trajectory optimization has developed a strategy using continuous-time trajectory optimization methods. In this paper, we demonstrate an exploration strategy using discrete-time trajectory optimization methods. We show significant improvements in performance speed, without a loss in optimization performance for the symplectic method. Therefore, we establish the importance in choice of integration methods. We show that the explicit Euler method can produce highly unstable, artificially ergodic trajectories, while the symplectic Euler is more robust to the time-discretization, producing trajectories that are stable and ergodic due to the structure-preserving nature of variational integrators.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant CMMI 1334609. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] B.D.O. Anderson and J.B. Moore. *Optimal Control: Linear Quadratic Methods*. Prentice Hall, 1990.

[2] T. Arbel and F.P. Ferrie. Viewpoint selection by navigation through entropy maps. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.*

[3] L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16, 1966.

[4] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[5] N. Cao, K.H. Low, and J.M. Dolan. Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms. In *Proceedings of the International Conference on Autonomous agents and Multi-agent systems*, pages 7–14, 2013.

[6] H. Choset. Coverage for robotics–a survey of recent results. *Annals of mathematics and artificial intelligence*, 31(1-4):113–126, 2001.

[7] H.J.S. Feder, J.J. Leonard, and C.M. Smith. Adaptive mobile robot navigation and mapping. *The International Journal of Robotics Research*, 18(7):650–668, 1999.

[8] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, volume 31 of *Springer Series in Computational Mathematics*. Springer, 2nd edition, 2006.

[9] J. Hauser. A projection operator approach to the optimization of trajectory functionals. In *IFAC World Congress*, volume 15, page 310, 2002.

[10] G.A. Hollinger and G. Sukhatme. Sampling-based motion planning for robotic information gathering. In *Robotics: Science and Systems*, 2013.

[11] E. Johnson J. Schultz and T.D. Murphey. Trajectory optimization in discrete mechanics. In Differential-Geometric Methods in Computational Multibody System Dynamics, In Press.

[12] E. Johnson, J. Schultz, and T. Murphey. Structured linearization of discrete mechanical systems for analysis and optimal control. *IEEE Transactions on Automation Science and Engineering*, 2014.

[13] K.H. Low, J.M. Dolan, and P. Khosla. Adaptive multi-robot wide-area exploration and mapping. In *Proceedings of the 7th International Joint Conference on Autonomous agents and Multiagent systems-Volume 1*, pages 23–30, 2008.

[14] W. Lu, G. Zhang, and S. Ferrari. An information potential approach to integrated sensor path planning and control. 2014.

[15] J. E. Marsden and M. West. Discrete mechanics and variational integrators. *Acta Numerica*, 10:357–514, 2001.

[16] G. Mathew and I. Mezić. Metrics for ergodicity and design of ergodic dynamics for multi-agent systems. *Physica D: Nonlinear Phenomena*, 240(4–5):432 – 442, 2011.

[17] Lauren M Miller, Yonatan Silverman, Malcolm A MacIver, and Todd D Murphey. Ergodic exploration of distributed information. *Transactions on Robotics*, Submitted.

[18] L.M. Miller and T.D. Murphey. Trajectory optimization for continuous ergodic exploration on the motion group SE(2). In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 4517–4522, Dec 2013.

[19] L.M. Miller and T.M. Murphey. Trajectory optimization for continuous ergodic exploration. In *American Control Conference (ACC)*, 2013.

[20] S. Ober-Blöbaum, O. Junge, and J. E. Marsden. Discrete mechanics and optimal control: an analysis. *Control, Optimisation and Calculus of Variations*, 17(2):322–352, 2011.

[21] D.O. Popa, A.C. Sanderson, R.J. Komerska, S.S. Mupparapu, D.R. Blidberg, and S.G. Chappel. Adaptive sampling algorithms for multiple autonomous underwater vehicles. In *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pages 108–118, June 2004.

[22] J.R. Spletzer and C.J. Taylor. Dynamic sensor planning and control for optimally tracking targets. *The International Journal of Robotics Research*, 22(1):7–20, 2003.

[23] P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *VMV*, volume 1, pages 273–280, 2001.