US Patent & Trademark Office
Patent Public Search | Text View

United States Patent Application Publication	20250256399
Kind Code	A1
Publication Date	August 14, 2025
Inventor(s)	SUN; Muchen et al.

KERNEL-BASED ERGODIC SEARCH

Abstract

According to one aspect, kernel-based ergodic search using a robot may include receiving a target distribution indicative of a desired ergodic search coverage, generating a kernel-based ergodic metric based on the target distribution and a candidate trajectory, generating a kernel-based ergodic gradient based on the kernel-based ergodic metric, generating a trajectory based on the kernel-based ergodic gradient, and implementing the trajectory for the robot.

Inventors: **SUN; Muchen (Evanston, IL), GAGGAR; Ayush (Evanston, IL), TRAUTMAN; Peter F. (Pebble Beach, CA), MURPHEY; Todd David (Evanston, IL)**

Applicant: **Honda Motor Co., Ltd.** (Tokyo, JP); **Northwestern University** (Evanston, IL)

Family ID: **96661485**

Appl. No.: **18/914929**

Filed: **October 14, 2024**

Related U.S. Application Data

us-provisional-application US 63553544 20240214

Publication Classification

Int. Cl.: **B25J9/16** (20060101); **G06F17/11** (20060101)

U.S. Cl.:

CPC **B25J9/1664** (20130101); **G06F17/11** (20130101);

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS [0001] This application claims the benefit of U.S. Provisional Patent Application, Ser. No. 63/553,544 (Attorney Docket No. H1240495US01) entitled “FAST ERGODIC SEARCH FOR EUCLIDEAN SPACE AND LIE GROUP”, filed on Feb. 14, 2024; the entirety of the above-noted application(s) is incorporated by reference herein.

BACKGROUND
[0002] Robots often need to search an environment driven by a distribution of information of interest. Examples

include search and rescue based on human annotated maps or aerial images, object tracking under sensory or motion uncertainty, and data collection in active learning. The success of such tasks depends on both the richness of the information representation and the effectiveness of the search algorithm. While advances in machine perception and sensor design have improved the quality of information representation, generating effective search strategies for the given information remains an open challenge.

[0003] Generally, ergodic search enables exploration of an information distribution while guaranteeing the asymptotic coverage of the search space. However, current methods typically have exponential computation complexity, which is undesirable, in the search space dimension and are generally restricted to Euclidean space or physical space. Therefore, higher dimensional searches currently require exponentially increasing computation resources. There are several limitations associated with a standard ergodic search: (1) a controller is limited with an infinitesimally small planning horizon; thus, it often requires an impractically long exploration period to generate good coverage; (2) it is costly to scale the standard ergodic metric to higher dimension spaces, (3) it is not trivial to generalize the metric to non-Euclidean spaces.

BRIEF DESCRIPTION

[0004] According to one aspect, a computer-implemented method for kernel-based ergodic search using a robot may include receiving, via a processor, a target distribution indicative of a desired ergodic search coverage, generating, via a metric generator, a kernel-based ergodic metric based on the target distribution and a candidate trajectory, generating, via a gradient generator, a kernel-based ergodic gradient based on the kernel-based ergodic metric, generating, via a controller, a trajectory based on the kernel-based ergodic gradient, and implementing, via a trajectory controller, the trajectory for the robot.

[0005] According to one aspect, a system for kernel-based ergodic search using a robot may include a memory storing one or more instructions and a processor executing one or more of the instructions stored on the memory to perform one or more acts, actions, and/or steps, such as receiving a target distribution indicative of a desired ergodic search coverage, generating a kernel-based ergodic metric based on the target distribution and a candidate trajectory, generating a kernel-based ergodic gradient based on the kernel-based ergodic metric, and generating a trajectory based on the kernel-based ergodic gradient. The system for kernel-based ergodic search using a robot may include a trajectory controller implementing the trajectory for the robot.

[0006] According to one aspect, a robot for kernel-based ergodic search may include a memory storing one or more instructions and a processor executing one or more of the instructions stored on the memory to perform one or more acts, actions, and/or steps, such as receiving a target distribution indicative of a desired ergodic search coverage, generating a kernel-based ergodic metric based on the target distribution and a candidate trajectory, generating a kernel-based ergodic gradient based on the kernel-based ergodic metric, and generating a trajectory based on the kernel-based ergodic gradient. The robot for kernel-based ergodic search may include a trajectory controller and one or more actuators implementing the trajectory for the robot.

Description

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] FIG. **1** is a flow diagram of a computer-implemented method for kernel-based ergodic search using a robot, according to one aspect.

[0008] FIG. **2** is a component diagram of a system or robot for kernel-based ergodic search, according to one aspect.

[0009] FIGS. **3**A-**3**E are graphs related to kernel-based ergodic search using a robot, according to one aspect.

[0010] FIG. **4** is an illustration of an example computing environment where one or more of the provisions set forth herein are implemented, according to one aspect.

[0011] FIG. **5** is an illustration of an example computer-readable medium or computer-readable device including processor-executable instructions configured to embody one or more of the provisions set forth herein, according to one aspect.

DETAILED DESCRIPTION

[0012] The following includes definitions of selected terms employed herein. The definitions include various examples and/or forms of components that fall within the scope of a term and that may be used for implementation. The examples are not intended to be limiting. Further, one having ordinary skill in the art will appreciate that the components discussed herein, may be combined, omitted, or organized with other components or organized into different architectures.

[0013] A “processor”, as used herein, processes signals and performs general computing and arithmetic functions. Signals processed by the processor may include digital signals, data signals, computer instructions, processor instructions, messages, a bit, a bit stream, or other means that may be received, transmitted, and/or detected. Generally, the processor may be a variety of various processors including multiple single and multicore processors and co-processors and other multiple single and multicore processor and co-processor architectures. The processor may include various modules to execute various functions.

[0014] A “memory”, as used herein, may include volatile memory and/or non-volatile memory. Non-volatile memory may include, for example, ROM (read only memory), PROM (programmable read only memory), EPROM (erasable PROM), and EEPROM (electrically erasable PROM). Volatile memory may include, for example, RAM (random access memory), synchronous RAM (SRAM), dynamic RAM (DRAM), synchronous DRAM (SDRAM), double data rate SDRAM (DDRSDRAM), and direct RAM bus RAM (DRRAM). The memory may store an operating system that controls or allocates resources of a computing device.

[0015] A “disk” or “drive”, as used herein, may be a magnetic disk drive, a solid-state disk drive, a floppy disk drive, a tape drive, a Zip drive, a flash memory card, and/or a memory stick. Furthermore, the disk may be a CD-ROM (compact disk ROM), a CD recordable drive (CD-R drive), a CD rewritable drive (CD-RW drive), and/or a digital video ROM drive (DVD-ROM). The disk may store an operating system that controls or allocates resources of a computing device.

[0016] A “bus”, as used herein, refers to an interconnected architecture that is operably connected to other computer components inside a computer or between computers. The bus may transfer data between the computer components. The bus may be a memory bus, a memory controller, a peripheral bus, an external bus, a crossbar switch, and/or a local bus, among others. The bus may also be a vehicle bus that interconnects components inside a vehicle using protocols such as Media Oriented Systems Transport (MOST), Controller Area network (CAN), Local Interconnect Network (LIN), among others.

[0017] A “database”, as used herein, may refer to a table, a set of tables, and a set of data stores (e.g., disks) and/or methods for accessing and/or manipulating those data stores.

[0018] An “operable connection”, or a connection by which entities are “operably connected”, is one in which signals, physical communications, and/or logical communications may be sent and/or received. An operable connection may include a wireless interface, a physical interface, a data interface, and/or an electrical interface.

[0019] A “computer communication”, as used herein, refers to a communication between two or more computing devices (e.g., computer, personal digital assistant, cellular telephone, network device) and may be, for example, a network transfer, a file transfer, an applet transfer, an email, a hypertext transfer protocol (HTTP) transfer, and so on. A computer communication may occur across, for example, a wireless system (e.g., IEEE 802.11), an Ethernet system (e.g., IEEE 802.3), a token ring system (e.g., IEEE 802.5), a local area network (LAN), a wide area network (WAN), a point-to-point system, a circuit switching system, a packet switching system, among others.

[0020] A “robot”, as used herein, may be a machine, such as one programmable by a computer, and capable of carrying out a complex series of actions automatically. A robot may be guided by an external control device or the control may be embedded within a controller. It will be appreciated that a robot may be designed to perform a task with no regard to appearance. Therefore, a “robot” may include a machine which does not necessarily resemble a human, including a vehicle, a device, a flying robot, a manipulator, a robotic arm, one or more robot systems, etc. Further, a “robot” may refer to any moving vehicle that is capable of carrying one or more human occupants and is powered by any form of energy. The term “robot” may include cars, trucks, vans, minivans, SUVs, motorcycles, scooters, boats, personal watercraft, and aircraft. In some scenarios, a motor vehicle includes one or more engines. Further, the term “robot” may refer to an electric vehicle (EV) that is powered entirely or partially by one or more electric motors powered by an electric battery. The EV may include battery electric vehicles (BEV) and plug-in hybrid electric vehicles (PHEV). Additionally, the term “robot” may refer to an autonomous vehicle and/or self-driving vehicle powered by any form of energy. The autonomous vehicle may or may not carry one or more human occupants.

[0021] A “robot system”, as used herein, may be any automatic or manual systems that may be used to enhance robot performance. Exemplary robot systems include a motor system, an autonomous driving system, an electronic stability control system, an anti-lock brake system, a brake assist system, an automatic brake prefill system, a low speed follow system, a cruise control system, a collision warning system, a collision mitigation braking system, an auto cruise control system, a lane departure warning system, a blind spot indicator system, a lane keep assist system, a navigation system, a transmission system, brake pedal systems, an electronic power steering system, visual devices (e.g., camera systems, proximity sensor systems), a climate control system, an electronic pretensioning system, a monitoring system, a passenger detection system, a suspension system, an audio system, a sensory system, among others.

[0022] FIG. **1** is a flow diagram of a computer-implemented method **100** for kernel-based ergodic search using a robot, according to one aspect. For example, the computer-implemented method **100** for kernel-based ergodic search using a robot may include receiving **102**, via a processor, a target distribution indicative of a desired ergodic search coverage. The target distribution may be a specification input to specify the desired ergodic search coverage. According to an example where the distribution is two-dimensional (2D), the distribution may be a 2D image. However, the distribution may often be in the form of a higher dimensional (e.g., greater than 2D) input, such as three-dimensional, four-dimensional, five-dimensional, six-dimensional, etc. The computer-implemented method **100** for kernel-based ergodic search using a robot may include generating **104**, via a metric generator, a kernel-based ergodic metric based on the target distribution and a candidate trajectory. The candidate trajectory may be initially provided as an initial input, but later be provided as a trajectory output by a controller, and iteratively refined, as described herein. The computer-implemented method **100** for kernel-based ergodic search using a robot may include generating **106**, via a gradient generator, a kernel-based ergodic gradient based on the kernel-based ergodic metric, generating **108**, via a controller, a trajectory based on the kernel-based ergodic gradient, and implementing **110**, via a trajectory controller, the trajectory for the robot. As discussed herein, the trajectory output by the controller may be fed back to the metric generator as the candidate trajectory, and the computer-implemented method **100** for kernel-based ergodic search using a robot may be iterated as desired.

[0023] The trajectory output by the controller may be a trajectory that generates the desired ergodic search coverage according to the distribution provided initially. Additionally, the trajectory may be represented such that a time history of the trajectory represents the desired ergodic search coverage according to the target distribution.

[0024] FIG. **2** is a component diagram of a system **200** or robot for kernel-based ergodic search, according to one aspect. The system **200** or robot for kernel-based ergodic search is described with reference to the computer-implemented method **100** for kernel-based ergodic search using the robot of FIG. **1**, and may include a processor, a metric generator, a gradient generator **214**, and a controller **216**. The metric generator, the gradient generator **214**, and the controller **216** may be implemented via the processor. The system **200** or robot for kernel-based ergodic search may include a memory **252**, a storage drive **254**, a communication interface **256**, a bus, and a trajectory controller **272**. The trajectory controller **272** may include one or more actuators **274**. The bus **258** may operably connect and enable computer communication between respective components of the system **200** or robot for kernel-based ergodic search. For example, the processor, the metric generator, the gradient generator **214**, the controller **216**, the memory **252**, the storage drive **254**, the communication interface **256**, the trajectory controller **272**, and the one or more actuators **274** may be operably connected to the bus.

Ergodic Theory

[0025] Generally, ergodic theory studies the connection between the time averaged and space averaged behaviors of a dynamical system. Originating in statistical mechanics, ergodic theory has now expanded to a full branch of mathematics with deep connections to other branches, such as information theory, measure theory, and functional analysis. For decision making, ergodic theory provides formal principles to reason over decisions based on the time and space averaged behaviors of the environment or of the agent itself. As used herein, “ergodicity”, may be utilized in the context of a search task given as the difference between the time averaged spatial statistics of the agent's trajectory and the target information distribution to perform a search. A quantitative measure of such difference is also introduced as a spectral multi scale coverage (SMC) metric (herein the “standard ergodic metric”), as well as a closed form model predictive controller with infinitesimally small planning horizon for both first order and second order linear dynamical systems.

[0026] Ergodic search has been applied to generate informative search behaviors in robotic applications, including multi-modal target localization, object detection, imitation learning, robotic assembly, and automated data collection for generative models. Additionally, the standard ergodic metric has been applied to non-search robotic applications, such as point cloud registration. Furthermore, ergodic search has also been extended to better satisfy other requirements from common robotic tasks, such as safety critical search, multi objective search, and time optimal search.

Formulations—Notations

[0027] The metric generator, implemented via the processor, may generate a kernel-based ergodic metric according to the following:

[0028] A state of a robot may be denoted as s∈S, where S is a bounded set within an n-dimensional Euclidean space. The state of the robot may be advantageously extended to Lie groups. The robot's motion may be governed by the following dynamics:

[00001] $\dot{s}(t)=f(s(t),u(t))$ (1)

[0029] where u(t)∈custom-character∈custom-character.sup.m is a control signal for a trajectory generated by the controller **216**, for example. The dynamics of function $f(\cdot,\cdot)$ is differentiable with respect to both s(t) and u(t). A probability density function may be defined over a bounded state space S and P(x):Scustom-character, which satisfies:

[00002] $\int_S P(x)dx = 1\text{and}P(x) \geq 0, \forall x \in S$ (2)

[0030] As used herein, a “trajectory” may be defined for s(t):[0, T]custom-characterS as a continuous mapping from time to a state in the bounded state space S.

[0031] As used herein, “trajectory empirical distribution” may be defined as follows: given a trajectory s(t):[0,T]custom-characterS for a robot, the empirical distribution of the trajectory s(t) may be represented as:

[00003] $C_s(x) = \frac{1}{T}\int_0^T \quad _{s(t)}(x)dt$ (3)

[0032] where δ.sub.s(t)(x) is a Dirac delta function. Here, the trajectory empirical distribution C.sub.s(x) has a valid probability density function. The trajectory empirical distribution C.sub.s(x) may be referred to as the time-averaged statistics of the trajectory since the trajectory empirical distribution C.sub.s(x) represents the likelihood of each region being visited by the robot. The trajectory empirical distribution C.sub.s(x) transforms the trajectory, as a function of time, to a spatial distribution in order to evaluate ergodicity.

Formulation—Ergodicity

[0033] A dynamic system is ergodic with respect to a spatial distribution if and only if the system visits any region of the space for an amount of time proportional to the integrated value of the target distribution over the region. Stated another way, a dynamic system is ergodic if the empirical distribution of its trajectory is the same as the spatial distribution, as the time horizon approaches infinity.

[0034] As used herein, an “ergodic system” may be defined as follows: given a spatial distribution P(x), a dynamical system, with its trajectory denoted as s(t), is ergodic with respect to the spatial distribution P(x) if and only if the following Equation (4) is satisfied:

[00004] $P(x) = \lim_{T\ .\text{fwdarw.}\ \infty} C_s(x), \forall x \in S$ (4)

[0035] In other words, an ergodic system's trajectory empirical distribution C.sub.s(x), at the limit of the infinite time horizon, is the same as the target distribution.

[0036] Regarding asymptotic coverage, if the spatial distribution P(x) has a non-zero density at any point of the state space S, an ergodic system, while systematically spending more time over regions with higher information density and less time over regions with less information density, will eventually visit any and/or every possible state in the state space S as the time approaches infinity.

[0037] However, standard ergodic systems are infeasible in practice for several reasons: (1) standard ergodic systems may only evaluate systems with infinitely-long time horizons; and (2) the trajectory empirical distribution C.sub.s(x) requires an evaluation of the Dirac delta function, which may only be evaluated in theory because of its infinite magnitude. For a robotic task, it may be desired to quantify how “ergodic” a finite-horizon trajectory is, given a spatial distribution P(x). By optimizing such a quantifiable metric as the objective for an optimal control or trajectory optimization problem, a robot may optimally explore the spatial distribution P(x) within a limited amount of time while knowing the asymptotic coverage property of an ergodic system, eventually guides the robot to cover the entire space. This motivates the derivation of the kernel-based ergodic metric, described in greater detail herein.

Formulation—Standard Ergodic Metric

[0038] Generally, an ergodic search generates a control signal for a dynamic system to maximize its ergodicity within a given spatial distribution P(x). Stated another way, an ergodic search controls a dynamic system to “match” the empirical distribution (e.g., trajectory empirical distribution C.sub.s(x)) of its trajectory with the spatial distribution P(x).

[0039] In other words, an ergodic metric measures the time averaged behavior of a dynamical system with respect to the spatial distribution P(x). A dynamic system is ergodic with respect to the spatial distribution P(x) if the system visits any region of the space for an amount of time proportional to the integrated value of the target distribution over the region. Optimizing the ergodic metric guides the robot to cover the whole search space asymptotically while investing more time in areas with higher information values. However, despite the theoretical advantages and tight connections to biological systems, current ergodic search methods are not suitable for many robotic tasks. The standard ergodic metric has an exponential computation complexity in the search space dimension, thereby limiting its applications in spaces with fewer than three dimensions. In this way, the standard ergodic metric is restricted to the Euclidean space. Moreover, common robotic tasks, such as vision-related or manipulation-related tasks, often require operations in non-Euclidean spaces, such as in the space of rotations or rigid body transformations.

[0040] The standard ergodic metric is formulated as a Sobolev norm through the Fourier transformation of the target distribution and the trajectory empirical distribution C.sub.s(x). The formula of the standard ergodic metric is described below.

[0041] As used herein, an “Fourier basis function” may be defined as follows: the standard ergodic metric assumes the robot operates in a n-dimensional rectangular Euclidean space, denoted as S=[0,L.sub.1]× . . . ×[0,L.sub.n]. The Fourier basis function f.sub.k(x): Scustom-character is defined as:

[00005] $f_k(x) = \frac{1}{h_k} \text{.Math.}_{i=1}^{n} \cos(\frac{k_i}{L_i} x_i)$ (5) [0042] where [0043] x=[x.sub.1, x.sub.2, . . . , x.sub.n]∈S; [0044] k=[k.sub.1, k.sub.2, . . . , k.sub.n]∈custom-character∈custom-character.sup.n; [0045] custom-character=[0, 1, . . . , K.sub.1]× . . . ×[0, 1, . . . , K.sub.n]; and [0046] h.sub.k is a normalization term such that the function space norm of each basis function is 1.

[0047] Thus, the set of Fourier basis functions forms a set of orthonormal basis functions. This means both the target spatial distribution and the trajectory empirical distribution C.sub.s(x) may be represented as the weighted sum of the Fourier basis functions.

[0048] As used herein, “standard ergodic metric” may be defined as follows: given an n-dimensional spatial distribution P(x) and a dynamical system with the trajectory s(t) over a finite time horizon [0, T], the standard ergodic metric, denoted as ε, is defined as:

[00006] $\mathcal{E}(P(x), s(t)) = \quad _k (p_k - c_k)^2$ (6)

[0049] where the sequences of {p.sub.k}custom-character and {c.sub.k}custom-character are the sequences of Fourier decomposition coefficients of the target distribution and trajectory empirical distribution C.sub.s(x), respectively:

[00007] $p_k = \int_S P(x) f_k(x) \mathrm{dx}$ (7) $c_k = \int_S C_s(x) f_k(x) dx = \frac{1}{T} \int_0^T f_k(s(t)) dt$ (8)

[0050] The sequence of {Λ.sub.k} is a convergent real sequence and is defined as:

[00008] $\quad _k = (1 + \text{ .Math. } k \text{ .Math. })^{-\frac{n+1}{2}}$ (9)

[0051] In practice, by choosing a finite number of Fourier basis functions, the standard ergodic metric ε may approximate ergodicity on a system with a finite time horizon. While a greater number of Fourier basis functions may lead to better approximations, the greater number of Fourier basis functions also requires more computational resources.

[0052] With the time horizon T and the number of Fourier basis functions approaching infinity, a dynamical system is globally optimal under the standard ergodic metric ε if and only if the system satisfies the original definition of an ergodic system from Equation (4).

[0053] The standard ergodic metric ε is a distance metric given by the Sobolev space norm of negative index [00009]$-\frac{n+1}{2}$:

[00010] $\mathcal{E}(P(x), s(t)) = \text{ .Math. } P(x) - C_s(x) \text{ .Math. }_{H^{-\frac{n+1}{2}}}^{2} .$ (10)

[0054] As discussed herein, one drawback to the standard ergodic metric ε of Equation (6) is the number of Fourier basis functions, which are associated with computational bottlenecks. Studies have revealed that a sufficient number of the basis functions for practical applications grows exponentially with the search space dimension, creating a significant challenge to apply the standard ergodic metric E in higher-dimensional spaces. Further, deriving the Fourier basis function in these spaces such as Lie groups is non-trivial, limiting the generalization of the standard ergodic metric ε.

Kernel-Based Ergodic Metric and Advantages

[0055] According to one aspect, the kernel-based ergodic metric may be formulated or expressed in terms of kernels. For example, the kernel-based ergodic metric may be formulated or expressed in terms of a delta kernel. Additionally, the kernel-based ergodic metric may be based on an L2 distance between a target information distribution and a spatial empirical distribution of a trajectory (e.g., candidate trajectory). Thus, the metric generator **212** may generate the kernel-based ergodic metric based on the target distribution and a candidate trajectory. According to one aspect, the candidate trajectory may be provided as an initial guess. For example, the metric generator **212** may estimate an initial candidate trajectory by approximating the candidate trajectory based on a nearest-neighbor approach.

[0056] This kernel-based ergodic metric provides the advantages and benefits of being asymptotically consistent with currently used standard ergodic search methods while being numerically efficient and guaranteeing linear complexity in the search space dimension (e.g., having linearly scaled computational requirements rather than exponentially scaled computational requirements), thereby enabling scaling beyond 2D spaces. Further, the kernel-based ergodic metric may be advantageously generalized to non-Euclidean spaces, such as Lie groups, as discussed herein. In this way, the kernel-based ergodic search provided herein enables an optimal balance between information maximization and uniform coverage (e.g., entropy maximization).

[0057] As discussed herein, the processor **210** may receive a target distribution indicative of a desired ergodic search coverage. Further, because the ergodic metric is a kernel-based ergodic metric, learning from demonstration (LfD) may be advantageously implemented using insufficient or unsuccessful demonstrations rather than successful demonstrations and without any sensor feedback. Ergodicity guarantees an asymptotic solution for a problem so long as a solution resides within the prior information distribution, and thus, a 100% success rate may be guaranteed. In this way, the kernel-based ergodic metric may be derived and ergodicity may be computed as a summation of an integrated likelihood of the trajectory s(t) within the spatial distribution P(x) and the uniformity of the trajectory s(t) measured with the kernel function.

Kernel-Based Ergodic Metric—Derivation

[0058] The metric generator **212** may generate the kernel-based ergodic metric based on an L2-norm distance formula for ergodicity, described in greater detail herein. In this regard, the Sobolev-norm and L2-norm formulas are asymptotically consistent as both converge to the definition of ergodic system in Equation (4), but the L2-norm formula leads to a more computationally efficient metric for ergodicity. The metric generator **212** may generate the kernel-based ergodic metric based on an L2 distance between the target information distribution (e.g., spatial distribution P(x)) and the trajectory empirical distribution C.sub.s(x) as follows:

[00011] $\int_S [P(x) - C_s(x)]^2 \mathrm{dx} = \quad (11) \quad \int_S [P(x)^2 - 2P(x)C_s(x) + C_s(x)^2]\mathrm{dx} = \quad (12)$

$\int_S P(x)^2 \mathrm{dx} - 2\int_S P(x)C_s(x)\mathrm{dx} + \int_S C_s(x)^2 \mathrm{dx} \quad (13)$

[0059] The first term of Equation (13) may be dropped since the norm of the target distribution ‖P(x)‖.sup.2=∫.sub.SP(x).sup.2dx does not depend on s(t) and is a constant given a target spatial distribution:

[00012]

$$\underset{s(t)}{\text{argmin}} \text{ .Math. } P(x) \text{ .Math. } - 2\int_S P(x)C_s(x)\mathrm{dx} + \int_S C_s(x)^2 \mathrm{dx} = \underset{s(t)}{\text{argmin}} - 2\int_S P(x)C_s(x)\mathrm{dx} + \int_S C_s(x)^2 \mathrm{dx} \quad (14)$$

[0060] Substituting the definition of trajectory empirical distribution C.sub.s(x) from Equation (3) to Equation (14). For the first term in Equation (14):

[00013] $\int_S P(x)C_s(x)\mathrm{dx} = \int_S P(x)[\tfrac{1}{T}\int_0^T \;{}_{s(f)}(x)\mathrm{dt}]\mathrm{dx} = \quad (15) \quad \tfrac{1}{T}\int_0^T [\int_S P(x) \;{}_{s(f)}(x)\mathrm{dx}]\mathrm{dt} = \quad (16)$

$\tfrac{1}{T}\int_0^T P(s(t))\mathrm{dt} \quad (17)$

[0061] Equations (16) and (17) are based on a property of the Dirac delta function. For the second term in Equation (14):

[00014] $\int_S C_s(x)^2 \mathrm{dx} = \int_S [\tfrac{1}{T}\int_0^T \;{}_{s(f)}(x)\mathrm{dt}]^2 \mathrm{dx} = \quad (18) \quad \tfrac{1}{T^2}\int_0^T \int_0^T [\int_S \;{}_{s(f_1)}(x) \;{}_{s(t_2)}(x)\mathrm{dx}]\mathrm{dt}_1\,\mathrm{dt}_2 \quad (19)$

$\tfrac{1}{T^2}\int_0^T \int_0^T \;(s(t_1), s(t_2))\mathrm{dt}_1\,\mathrm{dt}_2 \quad (20)$

[0062] where ϕ(x.sub.1, x.sub.2) is a delta kernel defined as:

[00015] $(x_1, x_2) = \begin{cases} +\infty, \mathrm{if} x_1 = x_2 \\ 0, \mathrm{otherwise} \end{cases} \quad (21)$

[0063] Substituting Equations (17) and (20) into Equation (14) results in the kernel-based ergodic metric as:

[00016] $(s(t)) = -\tfrac{2}{T}\int_0^T P(s(t))\mathrm{dt} + \tfrac{1}{T^2}\int_0^T \int_0^T \;(s(t_1), s(t_2))\mathrm{dt}_1\,\mathrm{dt}_2 \quad (22)$

[0064] Therefore, P(s(t)) represents the target spatial distribution and ϕ(s(t.sub.1),s(t.sub.2)) represents the kernel function (e.g., delta kernel, but any other kernel may be utilized, such as radial basis function kernels). The first term

[00017]$(e.g., -\tfrac{2}{T}\int_0^T P(s(t))\mathrm{dt})$

of Equation (22) facilitates achieving information maximization (e.g., information maximization element) while the second term

[00018]$(e.g., \tfrac{1}{T^2}\int_0^T \int_0^T \;(s(t_1), s(t_2))\mathrm{dt}_1\,\mathrm{dt}_2)$

facilitates achieving uniform coverage or entropy maximization (e.g., uniform coverage element). In this way, a kernel-based ergodic search may be provided to achieve an optimal balance between information maximization and uniform coverage (e.g., entropy maximization) and includes an information maximization element and a uniform coverage element.

[0065] In practice, the kernel ϕ(·,·) may be approximated in various forms and results hold as long as the kernel function asymptotically converges to a delta kernel. For example, Equation (22) may be specified with the squared exponential kernel function, the rational quadratic kernel function, γ-exponential kernel function, the Matérn class of kernel functions, etc. As will be described herein, the kernel-based ergodic metric is formulated as a Gaussian (e.g., squared exponential) kernel:

[00019] $$(x_1, x_2; \text{ .Math. }) = \frac{\exp(\frac{1}{2}(x_1 - x_2)^T \text{ .Math.}^{-1} (x_1 - x_2))}{\sqrt{(2\ \ )^n \det(\text{ .Math. })}} \quad (23)$$

[0066] Where Σ is a n-by-n covariance matrix of the kernel and ε.sub.ϕ(s(t)) is the kernel-based ergodic metric.

[0067] With the time horizon T approaching infinity and the kernel approaching a delta kernel of Equation (21), a dynamical system is globally optimal under the kernel-based ergodic metric ε.sub.ϕ(s(t)) of Equation (22) if and only if the system satisfies the original definition of an ergodic system from Equation (4).

[0068] With the time horizon T approaching infinity and the kernel approaching the delta kernel of Equation (21), the kernel-based ergodic metric ε.sub.ϕ(s(t)) of Equation (22) converges to the L2-distance of Equation (11). Since the L2-distance of Equation (11) is a convex metric with a global minima of 0, a dynamic system that is globally optimal under the kernel-based ergodic metric ε.sub.ϕ(s(t)) of Equation (22) must satisfy:

[00020] $$P(x) = C_s(x), \forall x \in S \quad (24)$$

[0069] which is equivalent to the definition of ergodic system in Equation (4).

[0070] The kernel-based ergodic metric ε.sub.ϕ(s(t)) of Equation (22) does not require the search space to be rectangular as required by the standard ergodic metric ε. Instead, the kernel-based ergodic metric ε.sub.ϕ(s(t)) merely requires the search space to be bounded.

Kernel-Based Ergodic Metric—Intuition

[0071] The formula for kernel-based ergodic metric ε.sub.ϕ(s(t)) of Equation (22) is the summation of two elements

[00021]$-\frac{2}{T}\int_0^T P(s(t))\mathrm{dt}and\frac{1}{T^2}\int_0^T \int_0^T \quad (s(t_1), s(t_2))\mathrm{dt}_1\,\mathrm{dt}_2$.

Minimizing the kernel-based ergodic metric ε.sub.ϕ(s(t)), which is equivalent to simultaneously minimizing the two elements, represents a balance between information maximization and uniform coverage.

[0072] For the first term, since P(x) is the probability density function of the target distribution, minimizing

[00022]$-\frac{2}{T}\int_0^T P(s(t))\mathrm{dt}$

drives the system to the state with the maximum likelihood within the spatial distribution P(x). For the second term in the kernel-based ergodic metric, minimizing the kernel-based ergodic metric ε.sub.ϕ(s(t)) drives the system to cover the search space uniformly. In other words, the trajectory s(t) that minimizes

[00023]$\frac{1}{T^2}\int_0^T \int_0^T \quad (s(t_1), s(t_2))\mathrm{dt}_1\,\mathrm{dt}_2$

uniformly covers the search space S.

[0073] As discussed, the kernel-based ergodic metric ε.sub.ϕ(s(t)) is the summation of the integrated likelihood of the trajectory s(t) within the spatial distribution P(x) and the uniformity of the trajectory s(t) within the search space measured by the kernel function. In other words, an ergodic system exhibits both information maximization behavior and uniform coverage behavior.

Kernel-Based Ergodic Metric—Optimal Kernal Parameter

[0074] The metric generator **212** may generate the kernel-based ergodic metric ε.sub.ϕ(s(t)) based on the target distribution and the candidate trajectory.

[0075] Since the delta kernel may be approximated by a Gaussian kernel from Equation (23), a parameter of the Gaussian kernel (e.g., a covariance matrix) may be optimized. According to one aspect, optimization may be based on a convergence property of the empirical distribution.

[0076] Assume s={s.sub.t} is a trajectory of a discrete dynamical system, with a total number of time steps N, where each state s.sub.t is an independent and identically distributed (IID) sample from the target distribution P(x). The following equation holds:

[00024] $$\lim_{N \text{ .fwdarw. } \infty} C_{\bar{s}}(x) = \lim_{N \text{ .fwdarw. } \infty} [\tfrac{1}{N} \text{ .Math.}_{t=1}^{N} \quad {}_{s(t)}(x)] = P(x), \forall x \in S \quad (25)$$

$$\lim_{N \text{ .fwdarw. } \infty} \text{ .Math. } \tfrac{d}{d\bar{s}}[\textstyle\int_S (C_s(x) - P(x))^2 \mathrm{dx}] \text{ .Math. } = 0 \quad (26)$$

[0077] The L2 distance may be replaced by the kernel-based ergodic metric ε.sub.ϕ(s(t)), but instead of a continuous time integral, a discrete Monte Carlo (MC) integral formula may be utilized:

[00025] $$\lim_{N \text{ .fwdarw. } \infty} \text{ .Math. } \tfrac{d}{d\bar{s}} J(\bar{s}) \text{ .Math. } = 0 \quad (27)$$

$$J(\bar{s}) = -\tfrac{1}{N} \text{ .Math.}_{t=1}^{N} P(s_t) + \tfrac{1}{N^2} \text{ .Math.}_{t_1=1}^{N} \text{ .Math.}_{t_2=1}^{N} \quad (s_{t_1}, s_{t_2}) \quad (28)$$

[0078] With reference to Equation (27), it appears that a set of IID samples from the target distribution are close to an optimum under the kernel-based ergodic metric ε.sub.ϕ(s(t)). Optimal kernel parameters may be selected by the metric generator **212** by minimizing the derivative of the samples with respect to the kernel-based ergodic metric ε.sub.ϕ(s(t)). This objective function may be defined for automatic kernel parameter selection:

[0079] As used herein, “kernel parameter selection objective” may be defined as follows: given a set of samples s={s.sub.i} from the target distribution, denote a kernel as ϕ(⋅,⋅; θ) with θ being the kernel parameter, the kernel parameter selection objective function for automatic kernel parameter selection is:

[00026] $J(\bar{s};\quad) = \text{.Math.}\ \frac{d}{d\bar{s}}(-\frac{1}{N}\ \text{.Math.}_{t=1}^{N}\ P(s_t) + \frac{1}{N^2}\ \text{.Math.}_{t_1=1}^{N}\ \text{.Math.}_{t_2=1}^{N}\quad (s_{t_1}, s_{t_2};\quad))\ \text{.Math.}^{2}$ (29)

[0080] In this way, the metric generator **212** may determine an optimal kernel parameter by minimizing the kernel parameter selection objective from Equation (29).

[0081] With a Gaussian kernel, as well as other kernel functions that are differentiable with respect to the parameter, the kernel parameter selection objective function is differentiable and may be optimized based on iterative optimization. If the metric generator **212** did not perform optimal kernel parameter selection based on the kernel parameter selection objective function from Equation (29), a neural network would be required to be generated based on a plurality of images that outputs a plurality of delta kernels and corresponding approximations along with a learning network that predicts optimal functions for use, thereby consuming a massive amount of computing resources.

Control—Kernel-Based Ergodic Metric

[0082] The kernel-based ergodic metric ε.sub.99 (s(t)) may be optimized when the trajectory s(t) is governed by a non-linear dynamical system. Even though the kernel-based ergodic metric ε.sub.99 (s(t)) has a double time-integral structure that is uncommon among standard optimal control formulas, the optimization may be solved through optimal control techniques, such as a linear-quadratic regulator (LQR).

Control—First Order Optimality Condition

[0083] The metric generator **212** may derive a first order optimality condition of the kernel-based ergodic metric ε.sub.ϕ(s(t)) for non-linear systems, which enables efficient trajectory optimization. The metric generator **212** may derive the first order optimality condition based on Pontryagin's maximum principle, thereby enabling an iterative application of LQR technique for efficient ergodic trajectory optimization.

[0084] The metric generator **212** may formulate the optimal control problem for the kernel-based ergodic metric ε.sub.ϕ(s(t)) and derive a first-order optimality condition for the optimal control problem based on the kernel-based ergodic metric ε.sub.ϕ(s(t)).

[0085] As used herein, “kernel ergodic control” may be defined as follows: given a target distribution P(x) and system dynamics {dot over (s)}(t)=f(s(t), u(t)), kernel ergodic control is defined as the following optimization problem:

[00027] $u^*(t) = \underset{u(t)}{\text{argmin}} J(u(t))$ (30) $J(u(t)) = \quad (s(t)) + \int_0^T l(s(t), u(t))\text{dt} + m(s(T))$ (31)

$s.t.s(t) = s_0 + \int_0^T f(s(\quad), u(\quad))d$ (32)

[0086] where l(·,·) and m(·) are the additional run-time cost and terminal cost of the task, respectively.

[0087] The derivation of the first order optimality condition for the above optimal control problem is based on the following:

[0088] Given a dynamical system {dot over (s)}(t)=f(s(t), u(t))∈custom-character.sup.n with the initial state s(0)=s.sub.0, denote v(t) as the perturbation on u(t) and z(t) as the resulting perturbation on s(t), z(t), and v(t) are governed by a linear dynamics with z(0)=0∈custom-character.sup.n:

[00028] $z(t) = A(t)z(t) + B(t)v(t)$ (33) $A(t) = D_1 f(s(t), u(t)), B(t) = D_2 f(s(t), u(t))$ (34)

[0089] The first-order condition for u(t) to minimize the objective of Equation (31) is:

[00029] $\quad(t)\ \ B(t) + b(t)\quad = 0, \forall t \in [0, T]$ (35) where $B(t) = \frac{d}{\text{du}} f(s(t), u(t)), b(t) = \frac{d}{\text{du}} l(s(t), u(t))$ (36) and

$\dot{}\,(t) = -A(t)\quad (t) - a(t)$ (37)$(T) = \text{Dm}(s(T))$ (38) $A(t) = \frac{d}{\text{ds}} f(s(t), u(t))$ (39)

$a(t) = -\frac{2}{T}\text{DP}(s(t)) + \frac{d}{\text{ds}} l(s(t), u(t)) + \frac{2}{T^2}\int_0^T D_1\quad (s(t), s(\quad))d$ (40)

[0090] where D.sub.1ϕ(·,·) denotes the derivative of ϕ with respect to its first argument.

[0091] Given a nominal control u(t) and the resulting state trajectory s(t), the Gateaux derivative of the kernel ergodic control objective from Equation (31) is the derivative of the objective in the direction of a perturbation v(t) on the control, which leads to a perturbation z(t) on the state, and may be computed as:

[00030] $\text{DJ}(u(t))\ \text{.Math.}\ v(t) = \int_0^T a(t)\quad z(t) + b(t)\quad v(t)\text{dt} + \quad z(T)$ (41) where $\quad = \text{Dm}(s(T))$ (42)

[0092] and a(t) and b(t) are defined in Equations (40) and (36), respectively.

[0093] The gradient generator may generate a kernel-based ergodic gradient based on the kernel-based ergodic metric ε.sub.ϕ(s(t)) and based on Equation (41) (e.g., the nominal control u(t), the resulting state trajectory s(t), the Gateaux derivative of the kernel ergodic control objective from Equation (31), the derivative of the objective in the direction of a perturbation v(t) on the control, the perturbation z(t) on the state, etc.).

Control—Iterative Optimal Control

[0094] The first-order optimality Equation (35) may be utilized to generate optimal control under the kernel-based ergodic metric ε.sub.ϕ(s(t)) by an iterative algorithm, described herein.

[0095] Equation (41) provides the closed-form expression for the descent direction of the kernel ergodic control objective and the perturbation is governed by a linear dynamics. Thus, instead of directly solving the optimization problem, the controller **216** may iteratively optimize the descent direction with a quadratic cost, which has a closed-form solution through LQR based on the kernel-based ergodic metric ε.sub.ϕ(s(t)) and the kernel-based ergodic gradient. Therefore, the sub-objective for finding the optimal descent direction J.sub.ζ(v(t)) is defined as:

[00031] $J\ \ (v(t)) = \mathrm{DJ}(u(t))\ .\mathrm{Math.}\ v(t) + \int_0^T z(t)\ \ Q(t)z(t) + v(t)\ \ R(t)v(t)\mathrm{dt} =$ (43)

$\int_0^T a(t)^T z(t) + b(t)\ \ v(t)\mathrm{dt} + \quad z(T) + \int_0^T z(t)\ \ Q(t)z(t) + v(t)\ \ R(t)v(t)\mathrm{dt}$ (44)

[0096] After finding the optimal descent direction in each iteration, the metric generator **212** may update the current control u(t) by choosing a step size for the descent direction, which may be fixed through tuning or adaptive through line search. The iterative optimal control process may be performed by the metric generator **212** according to Algorithm 1 below:

```
TABLE-US-00001 Algorithm 1 (kernel-based ergodic trajectory optimization)
1: procedure trajectory optimization (s.sub.0, ū(t))
    2: k ← 0, where custom-character k is the iteration index
    3: u.sub.k(t) ← ū(t)
    4: while a termination criterion is not met do
        5: simulate s.sub.k(t) given s.sub.0 and u.sub.k(t)
        6: v.sub.k(t) ← argmin.sub.v(t) J.sub.ζ(v(t)) custom-character Equation (44)
        7: find step size η custom-character fixed or apply line search
        8: u.sub.k+1(t) ← u.sub.k(t) + η .Math. u.sub.k(t)
        9: k ← k + 1
    10: end while
    11: return u.sub.k(t)
12: end procedure trajectory optimization
```

[0097] In this way, the controller **216** may generate a control signal for the trajectory s(t) based on the kernel-based ergodic gradient and the kernel-based ergodic metric. For example, the controller **216** may generate the control signal for the trajectory s(t) by iteratively solving the LQR problem based on Algorithm 1.

Control—Enhancing Optimization

[0098] According to one aspect, additional approaches to accelerate or enhance the computation of Algorithm 1 may be provided. For example, a first approach is a bootstrap method that provides an initial control ū(t) to the Algorithm 1 that is closer to the optimum. The second approach parallelizes the computation of the descent direction to speed up each iteration of the Algorithm 1 among multiple processors.

Control—Enhancing Optimization Via Bootstrap

[0099] The processor **210** may guide the initial control to roughly cover the target distribution. For example, to achieve this initial control, a trajectory tracking problem may be formulated for the initial control to track over an ordered set of samples from the target distribution. The order of the samples is determined by a rapid approximation of a traveling-salesman problem (TSP). Neither the solution of TSP nor trajectory tracking is required to be accurate. For example, the TSP solution may be approximated through a nearest-neighbor approach, which has a maximum quadratic complexity, and the trajectory tracking problem may be computed through any iterative optimization method for a few iterations, without strict convergence criterion.

Control—Enhancing Optimization Via Parallelization

[0100] The time integral term in the descent direction formula of Equation (40) is:

[00032] $\int_0^T D_1\ \ (s(t), s(\ \ ))d$ (45)

[0101] and is the summation of the derivative of kernel correlation between the state at time t and the whole trajectory. Since the evaluation of the kernel correlations in each time step are independent of one another, this integral may be computed in parallel. Similarly, the computation of the kernel ergodic control objective from Equation (31) may also be parallelized during the linear search in each iteration.

Lie Groups—Kernel-Based Ergodic Control

[0102] As discussed herein, the kernel-based ergodic metric ε.sub.ϕ(s(t)) may be advantageously generalized to other Riemannian manifolds or to non-Euclidean spaces, such as Lie groups (e.g., SE(3) special Euclidean group). Additionally, control derivations may be generalized to the Lie Groups.

Lie Groups—Preliminaries

[0103] A Lie group custom-character is a smooth manifold. Thus, any element on the Lie group custom-character locally resembles a linear space. But unlike other manifolds, elements in a Lie group custom-character also satisfy the four group axioms equipped with a composition operation: closure, identity, inverse, and associativity. Therefore, the Lie group custom-character may model non-Euclidean entities, such as rotation or rigid body transformation, while allowing analytical and numerical techniques in the Euclidean space to be applied. In particular, the special orthogonal group SO(3) and the special Euclidean group SE(3) may be used to model 3D rotation and 3D rigid body transformation (e.g., simultaneous rotation and translation), respectively.

[0104] As used herein, “SO(3) group” may be defined as follows: the special orthogonal group SO(3) is a matrix manifold, in which each element is a 3-by-3 matrix satisfying the following property:

[00033]$g\quad g = \mathrm{gg}\quad = I\mathrm{anddet}(g) = 1, \forall_g \in \mathrm{SO}(3) \subset \mathbb{R}^{3\times 3}$

[0105] where I is a 3-by-3 identity matrix. The composition operator for SO(3) is the standard matrix multiplication.

[0106] As used herein, “SE(3) group” may be defined as follows: the Special Euclidean group SE(3) is a matrix manifold. Each element of SE(3) is a 4-by-4 matrix that, when used as a transformation between two Euclidean space points, preserves the Euclidean distance between and the handedness of the points. Each element has the following structure:

[00034]$g = [\begin{matrix} R & t \\ 0 & 1 \end{matrix}], R \in \mathrm{SO}(3), t, 0 \in \mathbb{R}^3$

[0107] The composition operation in SE(3) is the standard matrix multiplication and has the following structure:

[00035] g_1 .Math. $g_2 = [\begin{matrix} R_1R_2 & R_1t_2+t_1 \\ 0 & 1 \end{matrix}]\quad (47)\quad g_1 = [\begin{matrix} R_1 & t_1 \\ 0 & 1 \end{matrix}], g_2 = [\begin{matrix} R_2 & t_2 \\ 0 & 1 \end{matrix}]\quad (48)$

[0108] The smooth manifold property of the Lie group custom-character means at every element in SO(3) and SE(3), a linear matrix space may be locally defined. This locally defined linear matrix space may be the tangent space of the Lie group custom-character.

[0109] As used herein, “tangent space” may be defined as follows: for an element g on a manifold custom-character, its tangent space custom-character is a linear space consisting of all possible tangent vectors that pass through g.

[0110] Each element in the tangent space custom-character may be considered as the time derivative of a temporal trajectory on the manifold g(t) that passes through the g at time t. Given the definition of a Lie group custom-character, the time derivative of such a trajectory is a vector.

[0111] As used herein, “Lie algebra” may be defined as follows: for a Lie group custom-character, the tangent space custom-character at its identity element custom-character is the Lie algebra of this group, denoted as custom-character.

[0112] Despite being a linear space, the tangent space custom-character on the Lie group custom-character and Lie algebra may still have non-trivial structures. For example, the Lie algebra of the SO(3) group is the linear space of skew-symmetric matrices. However, elements in Lie algebra may be expressed as a vector on top of a set of generators, which are the derivatives of the tangent element in each direction. Thus, Lie algebra elements may be represented in the standard Euclidean vector space. Elements between the Lie algebra and the standard Euclidean space may be transformed through isomorphisms, such as the hat operator and vee operator, discussed in greater detail herein.

[0113] As used herein, the “hat operator” may be defined as follows: the hat operator ({circumflex over (·)}) is an isomorphism from a n-dimensional Euclidean vector space to the Lie algebra with n-degrees of freedom:

[00036] $(\ .\overset{\wedge}{\mathrm{Math}}.\ ): \mathbb{R}^n \mapsto ; \hat{v} = \ .\mathrm{Math.}_{i=1}^{n}\ v_i E_i \in , v \in \mathbb{R}^n \quad (49)$

[0114] where E.sub.i is the i-th generator of the Lie algebra.

[0115] As used herein, the “vee operator” may be defined as follows: vee operator .sup.v(·):custom-character is the inverse mapping of the hat operator.

[0116] For the SO(3) group, the hat operator is defined as:

[00037] $\wedge = [\begin{matrix} 0 & - & {}_{e} & {}^{2} \\ {}_{3} & 0 & - & {}_{1} \\ - & {}_{2} & {}_{1} & 0 \end{matrix}], \quad \in \mathbb{R}^3 \quad (50)$

[0117] For the SE(3) group, the hat operator is defined as:

[00038] $\wedge = [\begin{matrix} \wedge & v \\ 0 & 0 \end{matrix}] \in \mathbb{R}^{4\times 4}, \quad = [\begin{matrix} \\ v \end{matrix}] \in \mathbb{R}^6, \quad , v \in \mathbb{R}^3 \quad (51)$

[0118] As used herein, “exponential map” may be defined as follows: an exponential map, denoted as exp: custom-character, maps an element from the Lie algebra to the Lie group custom-character.

[0119] As used herein, “logarithm map” may be defined as follows: a logarithm map, denoted as log: custom-character, maps an element from the Lie algebra to the Lie group custom-character.

[0120] The exponential and logarithm map for the SO(3) and SE(3) groups may be computed as desired. For example, the exponential map for the SO(3) group may be computed using Rodrigues' rotation formula.

[0121] As used herein, “adjoint” may be defined as follows: an adjoint of a Lie group element g, denoted as Ad.sub.g: custom-character, transforms the vector in one tangent space to another tangent space. Given two tangent spaces, custom-character and custom-character, from two elements of the Lie group custom-character, the adjoint enables the following transformation:

[00039] $v_1 = \mathrm{Ad}_{g_1^{-1} g_2}(v_2)$ (52)

[0122] Since the adjoint is a linear transformation, the adjoint may be represented as a matrix denoted as [Ad.sub.g]. The adjoint matrix for a SO(3) matrix is itself. The adjoint matrix for a SE(3) matrix is:

[00040] $[\mathrm{Ad}_g] = [\begin{matrix} R & \hat{t}R \\ 0 & R \end{matrix}] \in \mathbb{R}^{6\times 6}, g = [\begin{matrix} R & t \\ 0 & 1 \end{matrix}]$ (53)

Lie Groups—Kernels

[0123] The definition of a Gaussian kernel is built on top of a notion of a “distance” or a quadratic function of the “difference” between the two inputs. In this regard, to define a kernel in a Lie group, quadratic functions in Lie groups may be defined.

[0124] As used herein, “quadratic function” with reference to Lie groups may be defined as follows: given two elements g.sub.1, g.sub.2 on the Lie group custom-character, define the quadratic function as:

[00041] $C(g_1, g_2) = \frac{1}{2}$ .Math. $\log(g_2^{-1} g_1)$ .Math. $_M^2$ (54)

[0125] where M is the weight matrix and log denotes the Lie group logarithm.

[0126] The derivatives of the quadratic function, are as follows:

[00042] $D_1 C(g_1, g_2) = d\exp^{-1}(-\log(g_2^{-1} g_1))^{\top} M \log(g_2^{-1} g_1)$ (55)

$D_2 C(g_1, g_2) = -[\mathrm{Ad}_{g_1^{-1} g_2}]^T D_1 C(g_1, g_2)$ (56)

[0127] where d exp.sup.−1 denotes the trivialized tangent inverse of the exponential map.

[0128] Based on Equation (54) squared exponential kernel on Lie groups may be defined.

[0129] As used herein, “squared exponential kernel” with reference to Lie groups may be defined as follows:

[00043] $(g_1, g_2; \ , M) =$ .Math. $\exp(\frac{1}{2}$ .Math. $\log(g_2^{-1} g_1)$ .Math. $_M^2)$ (57)

Lie Groups—Probability Distributions

[0130] Generalizing Gaussian and Gaussian-mixture distributions to the Lie group as the target distribution are discussed herein. Although the following discussion relates to generalizing Gaussian and Gaussian-mixture distributions to the Lie group, other generalizations, such as the Cauchy distribution and Laplace distribution are contemplated. In any event, the discussion below follows a concentrated Gaussian formula, used for probabilistic state estimation on Lie groups.

[0131] As used herein, “Gaussian distribution” may be defined as follows: given a Lie group mean g∈ custom-character and a covariance matrix Σ whose dimension matches the degrees of freedom of the Lie group custom-character (e.g., and thus, the dimension of a tangent space custom-character on the group), a Gaussian distribution may be defined, denoted as custom-character (g, Σ), with the following probability density function:

[00044] $(g \mid \bar{g}, \ ) = \ (\log(\bar{g}^{-1} \circ g)$ .Math. $0, \ )$ (58)

[0132] where custom-character (0, Σ) is a zero-mean Euclidean Gaussian distribution in the tangent space custom-character of the mean custom-character.

[0133] Given the above definition, in order to generate a sample g˜custom-character(g, Σ) from the distribution, the processor **210** may generate a perturbation from the distribution of the tangent space ϵ˜ custom-character(0, Σ), which will perturb the Lie group mean to generate the sample:

[00045] $g = \bar{g} \circ \exp(\ ) \sim \ _g(\bar{g}, \ )$ (59) $\sim \ (0, \ )$ (60)

[0134] Following this relation, the Lie group Gaussian distribution and the tangent space Gaussian distribution share the same covariance matrix through the following equation:

[00046] $= \mathbb{E}[\epsilon\epsilon^T] =$ (61) $\mathbb{E}[\log(\bar{g}^{-1} \circ g)\log(\bar{g}^{-1} \circ g)^T]$ (62)

[0135] Since the optimal control formula requires the derivative of the target probability density function with respect to the state, the full expression of the probability density function and derive its derivative as follows:

[00047] $P(g) = \ (g \mid \bar{g}, \ )$ (63) $P(g) = \ (\log(\bar{g}^{-1} \circ g)$ .Math. $0, \ )$

$P(g) =$ .Math. $\exp(-\frac{1}{2}\log(\bar{g}^{-1} g)^{\top}$ .Math.$^{-1}$ $\log(\bar{g}^{-1} g))$

[0136] where η is the normalization term defined as:

[00048] $= \frac{1}{\sqrt{(2\ \ )^{n}\det(\ \ )}}$ (64)

[0137] The derivative of P(g) is:

[00049] $DP(g) = P(g)$ .Math. $-(\frac{d}{dg}\log(\bar{g}^{-1}g))^{\top}$ .Math.$^{-1}$ $\log(\bar{g}^{-1}g)$ (65)

[0138] where the derivative

[00050]$\frac{d}{dg}\log(\bar{g}^{-1}g)$

may be further expanded as:

[00051] $\frac{d}{dg}\log(\bar{g}^{-1}g) = d\exp(-\log(\bar{g}^{-1}g)$ .Math. $\frac{d}{dg}(\bar{g}^{-1}g))$ (66) $\frac{d}{dg}\log(\bar{g}^{-1}g) = d\exp(-\log(\bar{g}^{-1}g)$ (67)

[0139] where d exp denotes the trivialized tangent of the exponential map.

[0140] The formula of concentrated Gaussian distribution on Lie groups perturbs the Lie group mean on the right side of Equation (59). Another formula is to perturb the mean on the left side. The Lie group derivation of the kernel-based ergodic metric ε.sub.ϕ(s(t)) holds for both formulas. One difference between the two formulas is the frame in which the perturbation is applied.

Lie Groups—Dynamics

[0141] The trajectory controller **272** may control the system **200** or robot for kernel-based ergodic search to move or travel along the determined trajectory using the one or more actuators **274** and based on the control signal of the trajectory generated by the controller **216**, thereby implementing the trajectory for the robot, for example.

[0142] For example, given a trajectory evolving on the Lie group g(t): [0, T]custom-character, define trajectory dynamics in terms of a control vector field:

[00052] $\dot{g}(t) = f(g(t), u(t), t) \in$ (68)

[0143] In order to linearize the dynamics based on the trajectory optimization algorithm from Equation (41), model the dynamics through the left trivialization of the control vector field:

[00053] $(g(t), u(t), t) = g(t)^{-1} f(g(t), u(t), t) \in$ (69)

[0144] which enables the dynamics to be represented as:

[00054] $\dot{g}(t) = g(t)\ \ (g(t), u(t), t)$ (70)

[0145] Denote a perturbation on the control u(t) as v(t) and the resulting tangent space perturbation on the Lie group state as z(t)∈custom-character. In this regard, z(t) exhibits a similar linear dynamics as its Euclidean space counterpart:

[00055] $z(t) = A(t)z(t) + B(t)v(t)$ (71) $A(t) = D_1\ \ (g(t), u(t), t) - [\mathrm{Ad}_{\ \ (g(t), u(t), t)}]$ (72) $B(t) = D_2\ \ (g(t), u(t), t)$ (73)

[0146] Since the linearization of the dynamics is in the tangent space custom-character, this enables the processor **210** to perform Algorithm 1, and thus, optimize the control for kernel ergodic control on Lie groups.

[0147] FIGS. **3**A-**3**E are graphs related to kernel-based ergodic search using a robot, according to one aspect. In FIGS. **3**A-**3**E, an example objective function for kernel parameter selection is shown, as well as how different kernel parameters may influence a resulting ergodic trajectory. From FIGS. **3**A-**3**E, it may be seen that the kernel parameter is an adjustable parameter to generate coverage trajectories that balance behaviors between uniform coverage and seeking maximum likelihood estimation. Thus, a kernel parameter could be sub-optimal under the parameter selection objective, yet still generate viable trajectories for practitioners depending on the specific requirements of a task.

[0148] FIG. **3**A is a graph illustrating samples associated with a target or desired distribution associated with kernel-based ergodic search using a robot, according to one aspect.

[0149] FIG. **3**B is a graph illustrating a kernel parameter selection objective associated with kernel-based ergodic search using a robot, according to one aspect. For example, the kernel parameter selection objective function from Equation (29) is shown with the given samples from FIG. **3**A. In this example, the kernel parameter is the value of the diagonal entry in the covariance.

[0150] FIG. **3**C is a graph illustrating an “over-uniform” coverage behavior associated with kernel-based ergodic search using a robot, according to one aspect. For example, a sub-optimal kernel parameter may lead to such “over-uniform” coverage behavior.

[0151] FIG. **3**D is a graph illustrating an exemplary optimal coverage behavior associated with kernel-based ergodic search using a robot, according to one aspect. When the kernel parameter is optimized, the kernel parameter may generate an ergodic trajectory that allocates the time spent in each region to be proportional to the integrated probability density of the region, in accordance with Equation (29).

[0152] FIG. **3**E is a graph illustrating an “over-concentrated” coverage behavior associated with kernel-based ergodic search using a robot, according to one aspect. For example, another sub-optimal kernel parameter may lead to such “over-concentrated” coverage behavior.

[0153] FIG. **4** and the following discussion provide a description of a suitable computing environment to implement aspects of one or more of the provisions set forth herein. The operating environment of FIG. **4** is merely one example of a suitable operating environment and is not intended to suggest any limitation as to the scope of use or functionality of the operating environment. Example computing devices include, but are not limited to, personal computers, server computers, hand-held or laptop devices, mobile devices, such as mobile phones, Personal Digital Assistants (PDAs), media players, and the like, multiprocessor systems, consumer electronics, mini computers, mainframe computers, distributed computing environments that include any of the above systems or devices, etc.

[0154] Generally, aspects are described in the general context of “computer readable instructions” being executed by one or more computing devices. Computer readable instructions may be distributed via computer readable media as will be discussed below. Computer readable instructions may be implemented as program modules, such as functions, objects, Application Programming Interfaces (APIs), data structures, and the like, that perform one or more tasks or implement one or more abstract data types. Typically, the functionality of the computer readable instructions are combined or distributed as desired in various environments.

[0155] FIG. **4** illustrates a system **400** including a computing device **412** configured to implement one aspect provided herein. In one configuration, the computing device **412** includes at least one processing unit **416** and memory **418**. Depending on the exact configuration and type of computing device, memory **418** may be volatile, such as RAM, non-volatile, such as ROM, flash memory, etc., or a combination of the two. This configuration is illustrated in FIG. **4** by dashed line **414**.

[0156] In other aspects, the computing device **412** includes additional features or functionality. For example, the computing device **412** may include additional storage such as removable storage or non-removable storage, including, but not limited to, magnetic storage, optical storage, etc. Such additional storage is illustrated in FIG. **4** by storage **420**. In one aspect, computer readable instructions to implement one aspect provided herein are in storage **420**. Storage **420** may store other computer readable instructions to implement an operating system, an application program, etc. Computer readable instructions may be loaded in memory **418** for execution by the at least one processing unit **416**, for example.

[0157] The term “computer readable media” as used herein includes computer storage media. Computer storage media includes volatile and nonvolatile, removable, and non-removable media implemented in any method or technology for storage of information such as computer readable instructions or other data. Memory **418** and storage **420** are examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, Digital Versatile Disks (DVDs) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which may be used to store the desired information and which may be accessed by the computing device **412**. Any such computer storage media is part of the computing device **412**.

[0158] The term “computer readable media” includes communication media. Communication media typically embodies computer readable instructions or other data in a “modulated data signal” such as a carrier wave or other transport mechanism and includes any information delivery media. The term “modulated data signal” includes a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal.

[0159] The computing device **412** includes input device(s) **424** such as keyboard, mouse, pen, voice input device, touch input device, infrared cameras, video input devices, or any other input device. Output device(s) **422** such as one or more displays, speakers, printers, or any other output device may be included with the computing device **412**. Input device(s) **424** and output device(s) **422** may be connected to the computing device **412** via a wired connection, wireless connection, or any combination thereof. In one aspect, an input device or an output device from another computing device may be used as input device(s) **424** or output device(s) **422** for the computing device **412**. The computing device **412** may include communication connection(s) **426** to facilitate communications with one or more other devices **430**, such as through network **428**, for example.

[0160] Still another aspect involves a computer-readable medium including processor-executable instructions configured to implement one aspect of the techniques presented herein. An aspect of a computer-readable medium or a computer-readable device devised in these ways is illustrated in FIG. **5**, wherein an implementation **500** includes a computer-readable medium **502**, such as a CD-R, DVD-R, flash drive, a platter of a hard disk drive, etc., on which is encoded computer-readable data **504**. This encoded computer-readable data **504**, such as binary data including a plurality of zero's and one's as shown in **504**, in turn includes a set of processor-executable computer instructions **506** configured to operate according to one or more of the principles set forth herein. In this implementation **500**, the processor-executable computer instructions **506** may be configured to perform a method **508**, such as the computer-implemented method **100** of FIG. **1**. In another aspect, the processor-executable computer instructions **506** may be configured to implement a system, such as the system **200** of FIG. **2**. Many such computer-readable media may be devised by those of ordinary skill in the art that are configured to operate in accordance with the techniques presented herein.

[0161] As used in this application, the terms “component”, “module,” “system”, “interface”, and the like are generally intended to refer to a computer-related entity, either hardware, a combination of hardware and software, software, or software in execution. For example, a component may be, but is not limited to being, a process running on a processor, a processing unit, an object, an executable, a thread of execution, a program, or a computer. By way of illustration, both an application running on a controller and the controller may be a component. One or more components residing within a process or thread of execution and a component may be localized on one computer or distributed between two or more computers.

[0162] Further, the claimed subject matter is implemented as a method, apparatus, or article of manufacture using standard programming or engineering techniques to produce software, firmware, hardware, or any combination thereof to control a computer to implement the disclosed subject matter. The term “article of manufacture” as used herein is intended to encompass a computer program accessible from any computer-readable device, carrier, or media. Of course, many modifications may be made to this configuration without departing from the scope or spirit of the claimed subject matter.

[0163] Although the subject matter has been described in language specific to structural features or methodological acts, it is to be understood that the subject matter of the appended claims is not necessarily limited to the specific features or acts described herein. Rather, the specific features and acts described herein are disclosed as example aspects.

[0164] Various operations of aspects are provided herein. The order in which one or more or all of the operations are described should not be construed as to imply that these operations are necessarily order dependent. Alternative ordering will be appreciated based on this description. Further, not all operations may necessarily be present in each aspect provided herein.

[0165] As used in this application, “or” is intended to mean an inclusive “or” rather than an exclusive “or”. Further, an inclusive “or” may include any combination thereof (e.g., A, B, or any combination thereof). In addition, “a” and “an” as used in this application are generally construed to mean “one or more” unless specified otherwise or clear from context to be directed to a singular form. Additionally, at least one of A and B and/or the like generally means A or B or both A and B. Further, to the extent that “includes”, “having”, “has”, “with”, or variants thereof are used in either the detailed description or the claims, such terms are intended to be inclusive in a manner similar to the term “comprising”.

[0166] Further, unless specified otherwise, “first”, “second”, or the like are not intended to imply a temporal aspect, a spatial aspect, an ordering, etc. Rather, such terms are merely used as identifiers, names, etc. for features, elements, items, etc. For example, a first channel and a second channel generally correspond to channel A and channel B or two different or two identical channels or the same channel. Additionally, “comprising”, “comprises”, “including”, “includes”, or the like generally means comprising or including, but not limited to.

[0167] It will be appreciated that various of the above-disclosed and other features and functions, or alternatives or varieties thereof, may be desirably combined into many other different systems or applications. Also, various presently unforeseen or unanticipated alternatives, modifications, variations, or improvements therein may be subsequently made by those skilled in the art which are also intended to be encompassed by the following claims.

Claims

1. A computer-implemented method for kernel-based ergodic search using a robot, comprising: receiving, via a processor, a target distribution indicative of a desired ergodic search coverage; generating, via a metric generator, a kernel-based ergodic metric based on the target distribution and a candidate trajectory; generating, via a gradient generator, a kernel-based ergodic gradient based on the kernel-based ergodic metric; generating, via a controller, a trajectory based on the kernel-based ergodic gradient; and implementing, via a trajectory controller, the trajectory for the robot.

2. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, wherein the kernel-based ergodic metric is based on a delta kernel.

3. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, wherein the kernel-based ergodic metric is based on an L2 distance between the target distribution and a spatial empirical distribution of the candidate trajectory.

4. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, wherein the kernel-based ergodic metric includes an information maximization element and a uniform coverage element.

5. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, wherein the kernel-based ergodic metric is formulated as a Gaussian kernel.

6. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, comprising performing kernel parameter selection for the kernel-based ergodic metric based on a kernel parameter selection objective function by minimizing a derivative of one or more independent and identically distributed (IID) samples from the target distribution with respect to the kernel-based ergodic metric.

7. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, comprising generating the trajectory based on iteratively optimizing a descent direction of a kernel ergodic control objective associated with the kernel-based ergodic metric with a quadratic cost.

8. The computer-implemented method for kernel-based ergodic search using the robot of claim 7, wherein the iteratively optimizing the descent direction is based on a linear-quadratic regulator (LQR).

9. The computer-implemented method for kernel-based ergodic search using the robot of claim 1, wherein the kernel-based ergodic metric is generalized to a Lie group.

10. The computer-implemented method for kernel-based ergodic search using the robot of claim 9, wherein the Lie group is a special orthogonal group SO(3) or a special Euclidean group SE(3).

11. A system for kernel-based ergodic search using a robot, comprising: a memory storing one or more instructions; a processor executing one or more of the instructions stored on the memory to perform: receiving a target distribution indicative of a desired ergodic search coverage; generating a kernel-based ergodic metric based on the target distribution and a candidate trajectory; generating a kernel-based ergodic gradient based on the kernel-based ergodic metric; and generating a trajectory based on the kernel-based ergodic gradient; and a trajectory controller implementing the trajectory for the robot.

12. The system for kernel-based ergodic search using a robot of claim 11, wherein the kernel-based ergodic metric is based on a delta kernel.

13. The system for kernel-based ergodic search using a robot of claim 11, wherein the kernel-based ergodic metric is based on an L2 distance between the target distribution and a spatial empirical distribution of the candidate trajectory.

14. The system for kernel-based ergodic search using a robot of claim 11, wherein the kernel-based ergodic metric includes an information maximization element and a uniform coverage element.

15. The system for kernel-based ergodic search using a robot of claim 11, wherein the kernel-based ergodic metric is formulated as a Gaussian kernel.

16. A robot for kernel-based ergodic search, comprising: a memory storing one or more instructions; a processor executing one or more of the instructions stored on the memory to perform: receiving a target distribution indicative of a desired ergodic search coverage; generating a kernel-based ergodic metric based on the target distribution and a candidate trajectory; generating a kernel-based ergodic gradient based on the kernel-based ergodic metric; and generating a trajectory based on the kernel-based ergodic gradient; and a trajectory controller and one or more actuators implementing the trajectory for the robot.

17. The robot for kernel-based ergodic search of claim 16, wherein the processor performs kernel parameter selection for the kernel-based ergodic metric based on a kernel parameter selection objective function by minimizing a derivative of one or more independent and identically distributed (IID) samples from the target distribution with respect to the kernel-based ergodic metric.

18. The robot for kernel-based ergodic search of claim 16, wherein the processor generates the trajectory based on iteratively optimizing a descent direction of a kernel ergodic control objective associated with the kernel-based ergodic metric with a quadratic cost.

19. The robot for kernel-based ergodic search of claim 18, wherein the iteratively optimizing the descent direction is based on a linear-quadratic regulator (LQR).

20. The robot for kernel-based ergodic search of claim 16, wherein the kernel-based ergodic metric is generalized to a Lie group.