Task-space Kernels for Diverse Stein Variational MPC

Madhav Shekhar Sharma[1], Thomas Power[1], and Dmitry Berenson[1]

***Abstract*— Model predictive control (MPC) has a proven track record for delivering robust performance in many challenging control tasks, however non-linear system dynamics and non-convex costs can make these problems challenging to solve. By taking a probabilistic view and using approximate inference to solve the optimal control problem, we can improve the exploration of the search space. We use a non-parametric approximate inference method that finds diverse solutions using a kernel function. We propose Task-space Kernels which define similarity between solutions in task-relevant spaces and better capture spatio-temporal information. This helps us generate smooth, diverse, and low-cost trajectories for complex robotic problems. We demonstrate this strategy empirically on two multi-modal manipulation tasks with a 7DoF robot where our state, and end-effector space kernels achieve lower average costs and steps over the baselines.**

I. INTRODUCTION

Trajectory optimization and optimal control are widespread tools for synthesizing robot behaviors [1]–[3]. Many realistic systems exhibit non-linear dynamics and costs, such as those for avoiding complex obstacle geometries, typically leading to non-convex objectives. As a result, the performance of many optimization methods is highly dependent on initialization; a poor initialization can result in trajectories that collide with obstacles in the environment, or make poor progress toward the goal.

Recently, Stein Variational Model Predictive Control (SV-MPC) was proposed [4], which uses a variational inference view of optimal control to propose an algorithm that iteratively optimizes a distribution over low-cost controls while maximizing their diversity. This improves exploration of search space to give greater robustness to local-minima.

SV-MPC promotes diversity by using a kernel function that defines the similarity between different solutions and scales the particles to optimality. The gradient of the kernel is directly responsible for pushing the control particles apart. However, SV-MPC only uses variants of the Radial-basis function kernel which relies on an L2 difference between solutions. In this paper, we propose using more informative kernel functions that define diversity in more task-relevant space to improve control performance.

II. RELATED WORK

Early work framing planning as inference by Attias developed a message-passing algorithm for planning in discrete state and action spaces [5]. Since then, methods based on message-passing have been proposed for solving planning

[1]Authors are with the University of Michigan, Ann Arbor, MI, USA. {madhavss, tpower, dmitryb}@umich.edu. This work was supported in part by Wang Chu Chien-Wen Research Award.

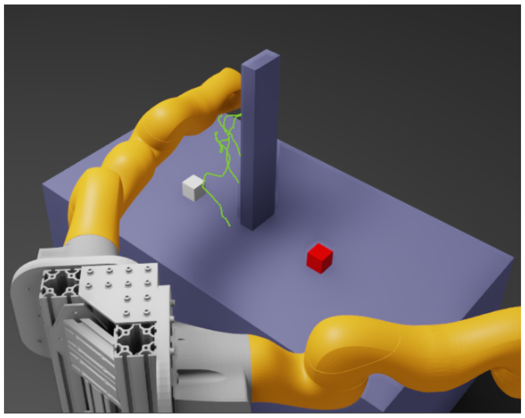


Fig. 1: An Episode of Pick and Place task. As depicted, our method is able to generate diverse collision-free paths, avoiding the infeasible local-minima, to pick the red target cube.

and control problems [6]–[9]. These methods use linearized Gaussian messages in cases when exact inference is not tractable, such as when the dynamics are non-linear.

Another popular class of methods are sample-based methods such as Model-predictive Path Integral (MPPI) [10] and the Cross-Entropy Method (CEM) [11]. Sample-based methods can result in non-smooth control actions; Watson and Peters propose using a Gaussian Process as the control sampling distribution [12], and Pinneri et al. proposed using colored noise [13], both in order to sample smoother control sequences.

Recently, Variational Inference (VI) has been used for planning and control as inference [4], [14]–[16]. These methods approximate a target distribution over trajectories with a simpler, parameterized distribution which minimizes the Kullback-Leibler divergence between the approximation and the target [17]. The performance is very dependent on the choice of parameterized distribution. Okada and Taniguchi use a Gaussian mixture [15]. Other recent work has learned a parameterization from data [18], [19]. Lambert et al. propose SV-MPC, which uses a particle representation [4]. This method uses Stein Variational Gradient Descent (SVGD) [20], which performs gradient descent on the particles to maximize their likelihood under the target distribution while also ensuring particle diversity, where diversity is determined by the choice of an appropriate kernel function. Methods using SVGD have also been applied to dual control and parameter identification [16] and constrained trajectory optimization [21]. Recent work, developed concurrently with our approach, has proposed kernels that are specific to trajectories [22] using path signatures. One key difference is that our method aims to use Kernels that define diversity in a different space to the parameter search space.

III. BACKGROUND

We consider the problem of Finite-horizon Stochastic Optimal Control with states $x \in R^{dx}$ & control $u \in R^{du}$. The trajectories of horizon H are represented as $\tau = (X, U)$, with $X = \{x_1, \cdots x_H\}$ and $U = \{u_0, \cdots u_{H-1}\}$. The controls U are parameterized by θ, this could be in the form of a feedback policy $u_t \sim \pi_\theta(u_t|x_t)$, however in this paper we optimize a parameterization of the open loop controls $u_t \sim \pi_\theta(u_t|t)$. Then, for a start state x_0, goal state x_G and an arbitrary cost function J, our objective is to find an optimal θ that minimizes the expected cost $E_{p(X|U)}[J(\tau)]$, with a known state transition probability $p(x_{t+1} \mid x_t, u_t)$ corresponding to stochastic dynamics $x_{t+1} \sim f(x_t, u_t) + \epsilon \quad, \epsilon \sim \mathcal{N}(0, R)$.

A. Variational Inference for Optimal Control

We formulate SOC as an inference problem, as in [7], [14], [15], [23], by introducing a binary random variable $\mathcal{O} \in \{0, 1\}$ representing optimality of a trajectory τ such that $p(\mathcal{O} = 1 \mid \tau) = \exp(-J(\tau))$. This recasts cost minimization as likelihood maximization. Rather than finding a single θ that maximizes this likelihood, we aim to compute the posterior $p(\theta \mid \mathcal{O} = 1) \propto \int p(\mathcal{O} = 1 \mid \tau)\, p_\theta(\tau)\, p(\theta) d\tau$, where

$$p_\theta(\tau) = \prod_{t=0}^{T-1} p(x_{t+1}|x_t, u_t)\pi_\theta(u_t|t) \tag{1}$$

is the distribution over trajectories with parameters θ and $p(\theta)$ is a prior over parameters.

In general, for systems with non-linear f and π, computing this posterior in closed form is intractable. Instead, we use Variational Inference to approximate $p(\theta|\mathcal{O} = 1)$ with a simpler distribution $q^*(\theta)$ chosen from a class of distributions $\mathcal{Q}$. We find $q^*(\theta)$ by minimizing the Kullback-Leibler (KL) divergence,

$$q^*(\theta) = \arg\min_{q \in \mathcal{Q}} \mathcal{D}_{KL}\left(q(\theta) \| p(\theta|\mathcal{O} = 1)\right). \tag{2}$$

We can rewrite the above minimization as

$$\begin{aligned}\mathcal{F}(q) &= -\mathbb{E}_{q(\theta)}\left[\log p(\theta|\mathcal{O} = 1)\right] - \mathcal{D}_{KL}\left(q(\theta) \| p(\theta)\right) \quad (3)\\ &= \mathbb{E}_{p_\theta(\tau)q(\theta)}\left[J(\tau)\right] - \mathbb{E}_{q(\theta)}[\log p(\theta)] - \mathcal{H}\left(q(\theta)\right), \quad (4)\end{aligned}$$

which is the *variational free energy*. We can interpret the first term in $\mathcal{F}(q)$ as promoting low-cost trajectories, the second term encourages consistency with the prior, and the third adding entropy to prevent the posterior from collapsing to a *maximum a posteriori* (MAP) solution.

B. Stein Variational Model Predictive Control

Choosing the distribution family $\mathcal{Q}$ of the candidate distribution $q(\theta)$ is often a difficult task, as we would like to combine flexibility with tractability. SV-MPC [4] is an algorithm for performing the minimization in equation (2) which uses a non-parametric representation for $q(\theta)$ using a variational inference technique called Stein Variational Gradient Descent (SVGD) [20]. SVGD uses non-parametric variational posterior $q(\theta) = \frac{1}{N}\sum_i^M \delta(\theta - \theta_i)$, where δ is the Dirac delta function and M is the number of particles. In the context of MPC, SVGD updates the set of particles iteratively via

$$\theta_i^{k+1} = \theta_i^k + \epsilon\phi^*(\theta_i), \tag{5}$$

$$\begin{aligned}\phi^*(\theta_i) = \frac{1}{n}\sum_{j=1}^{n} &\mathcal{K}(\theta_i, \theta_j)\nabla_{\theta_j}\log p(\theta_j|\mathcal{O} = 1) + \\ &\nabla_{\theta_j}\mathcal{K}(\theta_i, \theta_j),\end{aligned} \tag{6}$$

where $\mathcal{K}(\cdot, \cdot) \to \mathbb{R}$ is a positive definite Kernel function, and ϵ is a step-size parameter. We can understand this update intuitively as the first term being responsible for maximizing the log probability of all the particles, and the second term pushing the particles away from one another to prevent them from collapsing to the local MAP estimate.

Common choices for the kernel function $\mathcal{K}(\cdot, \cdot)$ are translation-invariant kernels such as the Radial Basis Function kernel or the Matérn kernel. Several recent studies have shown how SVGD is prone to variance collapse and vanishing repulsive forces during high-dimensional inference tasks [24]. Thus, choosing a suitable kernel $\mathcal{K}$ becomes important for high dimensional trajectory inference.

IV. METHOD

Our method builds upon the SV-MPC algorithm that optimizes diverse sets of low-cost solutions to the SOC problem. However, SV-MPC uses kernel functions to promote diversity that do not consider the inherent sequential nature of trajectories, nor the actual task state space X. In addition, kernels operating on high dimensional inputs, such as open loop control sequences, are often prone to diminishing repulsive force between particles. In this paper, we will illustrate how our approach of working with Task-space kernels improves VI-MPC performance with trajectory diversity and offers better, competitive control performance in manipulation tasks with a 7DoF robot.

A. Parameterization of Smooth Controls

In order to ensure that the control sequences we generate are smooth, we parameterize them through a set of support points $\{\theta_{t_i}, t_i\}_{i=0}^N$, with $N < H$, the horizon. We then interpolate these support points using an RBF kernel defined on time. Let $\mathcal{K}_N$ be the $\mathbb{R}^{N\times N}$ kernel matrix computed between all waypoint times $\{t_i\}_{i=1}^N$, and $\mathcal{K}_{Nt}$ be the $\mathbb{R}^N$ vector with i-th row given by $\mathcal{K}(t_i, t)$. Then we compute the control at time t as

$$u_t = h_\theta(t) = \mathcal{K}_{Nt}^T\mathcal{K}_N^{-1}\theta_N, \tag{7}$$

where θ_N is the $\mathbb{R}^{N\times d_u}$ matrix of waypoints. The control sampling distribution $u_t \sim \pi_\theta(u|t)$ is given by $u_t \sim h_\theta(t) + \omega, \quad \omega \sim \mathcal{N}(0, \Sigma)$.

B. Distance Metrics and Task-space Kernels

One of the most commonly used kernels used in SVGD is the RBF Kernel

$$\mathcal{K}(\theta_j, \theta) = \exp(-\frac{1}{h}\|\theta_j - \theta\|^2)$$

where for an optimization step with M particles, h is the kernel bandwidth. We follow prior work and use the median heuristic to choose the kernel bandwidth [25]

$$h = \frac{\mathtt{med}(\|\theta_j - \theta\|^2)}{2\log M + 1}$$

a) Factorized kernels: As previously mentioned, the gradient of the kernel which provides the repulsive forces can be very small during high-dimensional inference tasks. Lambert et al. [4] factorizes θ and consider a kernel that is the sum of kernels on the factors. We take a similar approach with a sliding window of size W

$$\mathcal{K}_{fact}(\theta_i, \theta_j) = \frac{1}{N-W}\sum_{k}^{N-W} \mathcal{K}(\theta_i^{k:k+W}, \theta_j^{k:k+W}) \quad (8)$$

b) Task-space kernels: Rather than use the L-2 norm in θ, we propose using a generalization of the RBF kernel as follows

$$\mathcal{K}(g(\theta_j), g(\theta)) = \exp(-\frac{1}{h}d(g(\theta_j), g(\theta)))$$

where $g(\theta)$ is a differentiable, deterministic function, and d is a distance metric. g maps θ into a different space in which we calculate the distance. We propose mapping θ to the sequence of states X in order to calculate distance, as this more explicitly encourages exploration of the state space. To make the mapping deterministic, we use the deterministic parts of the dynamics and control parameterization, f and h. By appropriately composing f and h we can compute $X = g(\theta)$. This is because part of our stochastic state transition function that gives $x_{t+1} = f(x_t, \theta_t) + \epsilon$. With such a formulation, we can work directly with deterministic X as a function of parameters θ [26] to have a generalized *task-space* kernel as

$$\mathcal{K}(X_i, X_j) = \exp(-\frac{1}{h}d(X_i, X_j)))$$

If the state space x consists of a robot configuration we use forward kinematics to map to a task-space pose $(p, \mathbf{R})$. In addition to mapping θ to the state-space, we can additionally map θ to other tasks relevant spaces.

c) Metrics for SE(3).: When defining a kernel on poses we must define a distance metric for SE(3). Consider two rotations $(\mathbf{R}_i, \mathbf{R}_j \in \mathbf{SO(3)})$ and the end-effector position $(p_i, p_j \in \mathbf{R}^3)$. For p_i, p_j, we use L-2 distance. However, for our 3D rotations $(\mathbf{R}_i, \mathbf{R}_j)$, we use the following distance function from [27] that respects both the axioms of a valid metric and the inherent topology of $\mathbf{SO(3)}$:

$$\phi : SO(3) \times SO(3) \to \mathbb{R}\left[0, 2\sqrt{2}\right]$$
$$\phi(\mathbf{R_1}, \mathbf{R_2}) = \|\mathbf{I} - \mathbf{R_1}\mathbf{R_2}^T\|_F$$

To calculate an overall pose distance we combine the position and rotation distances via

$$d_{pose}(p_i, \mathbf{R}_i, p_j, \mathbf{R}_j) = \alpha\phi(\mathbf{R}_i, \mathbf{R}_j)^2 + \beta||p_i - p_j||_2^2, \quad (9)$$

where $\alpha, \beta \in \mathbb{R}$.

d) Fréchet Metric: In order to compute more informative distances between two paths in state or pose space, we use the Fréchet distance, a popular approach for measuring similarity between finite curved paths [28]. It measures the gap between the two most distant points for a pair of paths given the best alignment of points along the path with their pairs. This idea of working with the most distant point in an optimal path alignment makes this metric extract features that are both spatial and temporally relevant. It has also been applied successfully in a wide range of trajectory analysis, classification, and optimization problems [29], [30].

In practice, we work with the discrete Fréchet distance (DFD) [31] which is defined for trajectories $X, Y \in \mathbb{R}^{H \times dx}$ and the set $\mathcal{A}$ of all of their possible alignments as

$$\mathcal{D}_{\mathit{Frechet}}(X, Y) = \min_{A \in \mathcal{A}} \max_{(i,j) \in A} d(x_i, y_j),$$

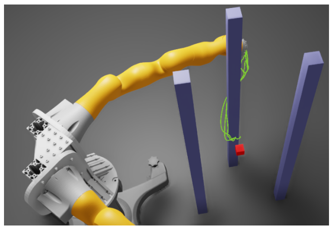
where $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a distance metric between the path vectors, typically the L-2 norm, but for paths in $\mathbf{SO(3)}$, it could be also be replaced by a metric such as ϕ. The above formulation is computed via dynamic programming in $O(H^2)$ time. Note also the non-smoothness within this formulation that hinders differentiability of a kernel in SVGD that uses metric as $d(\cdot, \cdot)$. To account for this, we'll utilize the smooth maximum (LogSumExp) defined as $LSE(x)_i = \log \Sigma_j(x_{ij})$ for approximating the min, max operations in the DFD as follows, with γ as the smoothening sensitivity.

$$\mathcal{D}_{\mathit{SF}}(X, Y) = -\gamma LSE(-\frac{\phi_{max}}{\gamma}),$$

where, $\phi_{max} = \gamma LSE(d(x_i, y_j)/\gamma)$. This is a similar approach to the method discussed in [32]. It is also useful to discuss $\nabla\mathcal{D}_{\mathit{Frechet}}$ as we are required to take the gradient of the kernel $\nabla(\cdot, \cdot)$ that uses this metric. The analytical negative gradient of the DFD points us towards the furthest point on our reference path. A step in this direction moves our maximum deviation point on the candidate path closer to the reference one, thus decreasing the distance.

C. Stein Variational MPC with Task-space Kernels

In this section, we will present our Task-Space Kernels that build upon the previous Stein MPC [4] framework to improve performance by optimizing low-cost, smooth and diverse trajectories in high-dof, multi-modal robot manipulation problems. We utilize the metrics discussed in the previous section to formulate a range of trajectory kernels through which we incorporate useful spatio-temporal information into efficient inference for planning. We first use two kernels


(a) Reaching Task

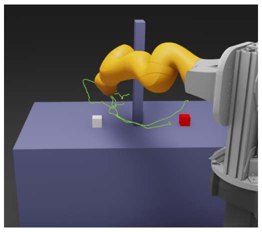

(b) Pick and Place Task

Fig. 2: End effector paths visualized with the Task-space kernels: Left setup with $\mathcal{K}_{FEE}$ and Right setup with $\mathcal{K}_q$.
Reach task has three tall obstacles that the Robot has to move its end-effector around the narrow gaps to reach the goal in the center. Pick and place task consists of a table with a slab in the middle and a target cube placed on one side. The objective here is to reach the cube while avoiding obstacle contact, perform the picking maneuver and plan again to the dropping position across the table.

defined on the state space

$$\mathcal{K}_q(X_i, X_j) = \exp(-\frac{1}{h}d(X_i, X_j)) \tag{10}$$

$$\mathcal{K}_{Fq}(X_i, X_j) = \exp(-\frac{1}{h}\mathcal{D}_{SF}(X_i, X_j)). \tag{11}$$

$\mathcal{K}_q$ uses the L-2 distance, whereas $\mathcal{K}_{Fq}$ uses the smooth Fréchet distance. We also define two kernels in the end-effector pose space. Consider pose trajectories $P_i, P_j \in \mathbf{SE(3)}^H$, we have

$$\mathcal{K}_{EE}(P_i, P_j) = \frac{1}{H}\sum_{k}^{H} \exp(-\frac{1}{h}d_{pose}(P_i^k, P_j^k)) \tag{12}$$

$$\mathcal{K}_{FEE}(P_i, P_j) = \exp(-\frac{1}{h}\mathcal{D}_{SF}(P_i, P_j)). \tag{13}$$

$\mathcal{K}_{EE}$ is a kernel on pairs of poses. $\mathcal{K}_{FEE}$ uses the smooth Fréchet distance, and inside the calculation uses the pose distance, ϕ, between pairs of poses. Now, with the kernels and the trajectory representation discussed above, we can adapt to the SV-MPC framework to have a kernelized sequential update to the Bayesian MPC posterior. We consider the relative probabilistic weights of our particles as approximation of the posterior and choose the highest weighted one ($\theta_t = \theta_{i*}$) to get the next control action u_{t+1}.

V. Results and Discussion

Here we evaluate our proposed kernels on two tasks with a 7DoF robot manipulator. The system's state space is the robot's joint configuration $q \in \mathbb{R}^7$. Controls are the joint actions $\dot{q}$ given by Euler integration as $q_{t+1} = q_t + \dot{q}dt$ and $dt = 0.67$. We also compare our method to MPPI, representing the state-of-the-art sampling-based SOC methods, and SV-MPC, the method that we build upon. All algorithms were written in PyTorch and the GPU-accelerated IsaacSim was used for parallel computation of rollouts and visualization. All hyper-parameters for the controllers are kept the same wherever possible to minimize variance of experimental setup.

The tasks in Fig 2 are configured to have natural local-minima traps, with the deceiving optimal trajectory at the side of the obstacles further from the robot. There is inherent multi-modality throughout the reach task while the second task is challenging due to its pose constraints for picking and placing and the need for long-horizon planning with obstacles.

We observe that in both tasks, all the kernel-based SV-MPC methods complete the task successfully and outperform MPPI, with lower costs although sometimes at the expense of slightly more steps. For the reach task, as we note from Table I, our pose kernels ($\mathcal{K}_{FEE}, \mathcal{K}_{EE}$) and state kernels ($\mathcal{K}_q, \mathcal{K}_{Fq}$) perform the reaching maneuver with lower average cost and within competent steps. We attribute this success to their behavior of generating a diverse spread of low-cost trajectories that converge to collision-free sets quite early.

Table II shows results when evaluating our kernel methods on the pick and place task. Once again, we observe that our proposed kernels outperforms MPPI. MPPI either collided with the slab, struggled to discover the feasible mode or timed out trying to stabilize the gripper over the target cube.

The Frechet kernel in EE-pose space $\mathcal{K}_{FEE}$ had the overall best performance in terms of lower nominal trajectory cost and steps taken for both the sub-tasks. The EE-state space kernel $\mathcal{K}_q$ completed the goal maneuver more often than all other methods, only with slightly higher number of steps and accumulated cost.

These were closely followed by the rest of the task-space kernels ($\mathcal{K}_{Fq}, \mathcal{K}_{EE}$) that were just as competent, if not better, than the baseline SV-MPC factorized kernel. MPPI on the other hand struggled to find the other modes and had poor trajectories that often collapsed close to the slab, or over the target locations.

TABLE I: Performance and comparison against baselines for the two tasks. We report both the average and the standard deviation of cost and steps taken to success. Table I shows results for the reaching task while Table II below reports results for the pick and place tasks, as shown in Fig. 2. We evaluate with 10 randomly sampled start configurations of the robot.

Method	SR	Cost $\mu(\sigma)$	Steps $\mu(\sigma)$
MPPI	80	84.9 (74.9)	**61.8 (8.6)**
$\mathcal{K}_{fact}$	**100**	30.6 (3.7)	84.3 (6.2)
$\mathcal{K}_q$	**100**	28.7 (3.5)	86.8 (15.5)
$\mathcal{K}_{EE}$	**100**	30.4 (3.4)	82.3 (7.1)
$\mathcal{K}_{Fq}$	**100**	30.7 (3.9)	85.2 (12.1)
$\mathcal{K}_{FEE}$	**100**	**28.3 (2.3)**	**77.7 (11.2)**

Method	SR	Cost $\mu(\sigma)$		Steps $\mu(\sigma)$	
		Picking	*Placing*	*Picking*	*Placing*
MPPI	-	-	-	-	-
$\mathcal{K}_{fact}$	70	**96.8 (12.1)**	49.6 (9.1)	233.6 (30.1)	184.9 (21.2)
$\mathcal{K}_q$	**85**	108.1 (12.3)	47.5 (11.4)	219.1 (27.1)	189.9 (43.8)
$\mathcal{K}_{EE}$	80	98.4 (9.3)	55.3 (9.5)	233.6 (10)	213.3 (24.4)
$\mathcal{K}_{Fq}$	80	98.9 (9.1)	51.7 (23.6)	224.6 (22.8)	176.8 (29.6)
$\mathcal{K}_{FEE}$	70	99 (11.4)	**42.7 (12.4)**	**194.7 (34.9)**	**161.9 (28.1)**

TABLE II: In both the tables, SR is Success Rate %age accross the trials, $\mathcal{K}_{fact}$ is the *factorized* kernel from SV-MPC, while the rest of the methods are our task-space kernels.

For future work, we plan to extend our task-space kernel framework with more sophisticated methods that are topologically able to capture path diversity for planning in cluttered, multi-modal robotic environments. Finally, we would like to thank the members of the ARM lab for their helpful discussions and insights.

References

[1] R. Bonalli, A. Cauligi, A. Bylard, and M. Pavone, "Gusto: Guaranteed sequential trajectory optimization via sequential convex programming," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2019, pp. 6741–6747.

[2] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *Int. J. Rob. Res.*, vol. 33, no. 9, pp. 1251–1270, 2014.

[3] C. Brasseur, A. Sherikov, C. Collette, D. Dimitrov, and P.-B. Wieber, "A robust linear mpc approach to online generation of 3d biped walking motion," in *Proc. 15th IEEE-RAS Int. Conf. Humanoid Robots*, 2015, p. 595–601.

[4] A. Lambert, A. Fishman, D. Fox, B. Boots, and F. Ramos, "Stein variational model predictive control," in *Proc. Conf. Robot Learn.*, 2020.

[5] H. Attias, "Planning by probabilistic inference," in *Proc. 9th Int. Workshop on Artificial Intelligence and Statistics*, 2003, pp. 9–16.

[6] M. Toussaint and A. Storkey, "Probabilistic inference for solving discrete and continuous state markov decision processes," in *Proc. Int. Conf. Mach. Learn.*, 2006, p. 945–952.

[7] K. Rawlik, M. Toussaint, and S. Vijayakumar, "On stochastic optimal control and reinforcement learning by approximate inference," in *Robot.: Sci. Syst.*, 2013.

[8] J. Watson, H. Abdulsamad, and J. Peters, "Stochastic optimal control as approximate input inference," in *Proc. Conf. Robot Learn.*, vol. 100, 2020, pp. 697–716.

[9] K. C. Rawlik, "On probabilistic inference approaches to stochastic optimal control," Ph.D. dissertation, The University of Edinburgh, 2013.

[10] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Trans. Robot.*, vol. 34, no. 6, p. 1603–1622, Dec 2018.

[11] M. Kobilarov, "Cross-entropy motion planning," *Int. J. Rob. Res.*, vol. 31, no. 7, pp. 855–871, 2012.

[12] J. Watson and J. Peters, "Inferring smooth control: Monte carlo posterior policy iteration with gaussian processes," in *Proc. Conf. Robot. Learn.*, 2023, pp. 67–79.

[13] C. Pinneri, S. Sawant, S. Blaes, J. Achterhold, J. Stueckler, M. Rolinek, and G. Martius, "Sample-efficient cross-entropy method for real-time planning," in *Proc. Conf. Robot Learn.*, vol. 155, 2021, pp. 1049–1065.

[14] Z. Wang, O. So, J. Gibson, B. Vlahov, M. Gandhi, G.-H. Liu, and E. Theodorou, "Variational Inference MPC using Tsallis Divergence," in *Robot.: Sci. Syst.*, 2021.

[15] M. Okada and T. Taniguchi, "Variational inference mpc for bayesian model-based reinforcement learning," in *Proc. Conf. Robot Learn.*, 2020, pp. 258–272.

[16] L. Barcelos, A. Lambert, R. Oliveira, P. Borges, B. Boots, and F. Ramos, "Dual Online Stein Variational Inference for Control and Dynamics," in *Robot.: Sci. Syst.*, 2021.

[17] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *J. Am. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, 2017.

[18] T. Power and D. Berenson, "Variational inference mpc using normalizing flows and out-of-distribution projection," in *Robot.: Sci. Syst.*, 2022.

[19] J. Sacks and B. Boots, "Learning sampling distributions for model predictive control," in *Proc. Conf. Robot. Learn.*, 2023, pp. 1733–1742.

[20] Q. Liu and D. Wang, "Stein variational gradient descent: A general purpose bayesian inference algorithm," in *Proc. Int. Conf. Neural Information Processing Systems*, 2016, p. 2378–2386.

[21] T. Power and D. Berenson, "Constrained stein variational trajectory optimization," 2023, arxiv.2308.12110.

[22] L. Barcelos, T. Lai, R. Oliveira, P. Borges, and F. Ramos, "Path signatures for diversity in probabilistic trajectory optimisation," 2023, arxiv.2308.04071.

[23] M. Toussaint, "Robot trajectory optimization using approximate inference," in *Proc. Int. Conf. Mach. Learn.*, 2009, p. 1049–1056.

[24] J. Ba, M. A. Erdogdu, M. Ghassemi, S. Sun, T. Suzuki, D. Wu, and T. Zhang, "Understanding the variance collapse of SVGD in high dimensions," in *International Conference on Learning Representations*, 2022.

[25] D. Garreau, W. Jitkrittum, and M. Kanagawa, "Large sample analysis of the median heuristic," 2018.

[26] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[27] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, pp. 155–164, 2009.

[28] M. M. Fréchet, "Sur quelques points du calcul fonctionnel," *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, vol. 22, no. 1, pp. 1–72, 1906.

[29] E. Sriraghavendra, K. Karthik, and C. Bhattacharyya, "Fréchet distance based approach for searching online handwritten documents," in *Ninth international conference on document analysis and recognition (ICDAR 2007)*, vol. 1. IEEE, 2007, pp. 461–465.

[30] R. M. Holladay and S. S. Srinivasa, "Distance metrics and algorithms for task space path optimization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 5533–5540.

[31] H. Alt and M. Godau, "Computing the fréchet distance between two polygonal curves," *International Journal of Computational Geometry & Applications*, vol. 5, no. 01n02, pp. 75–91, 1995.

[32] K. Takeuchi, M. Imaizumi, S. Kanda, Y. Tabei, K. Fujii, K. Yoda, M. Ishihata, and T. Maekawa, "Fréchet kernel for trajectory data analysis," in *Proceedings of the 29th International Conference on Advances in Geographic Information Systems*, 2021, pp. 221–224.