Hierarchical Policy Blending as Inference for Reactive Robot Control

Kay Hansel[1], Julen Urain[1], Jan Peters[1−4] and Georgia Chalvatzaki[1,3]

Abstract— Motion generation in cluttered, dense, and dynamic environments is a central topic in robotics, rendered as a multi-objective decision-making problem. Current approaches trade-off between safety and performance. On the one hand, reactive policies guarantee a fast response to environmental changes at the risk of suboptimal behavior. On the other hand, planning-based motion generation provides feasible trajectories, but the high computational cost may limit the control frequency and, thus, safety. To combine the benefits of reactive policies and planning, we propose a hierarchical motion generation method. Moreover, we employ probabilistic inference methods to formalize the hierarchical model and stochastic optimization. We realize this approach as a weighted product of stochastic, reactive expert policies, where planning is used to adaptively compute the optimal weights over the task horizon. This stochastic optimization avoids local optima and proposes feasible *reactive* plans that find paths in cluttered and dense environments. Our extensive experimental study in planar navigation and 7DoF manipulation shows that our proposed hierarchical motion generation method outperforms both myopic reactive controllers and online re-planning methods. Additional material available at `https://sites.google.com/view/hipbi`.

Fig. 1: A sequence of the reactive motion of a 7DoF manipulator robot. The robot starts moving from the orange box toward the green box. Our proposed method enables a reactive motion that avoids collisions with the grey obstacle and overcomes local minima resulting from multiple constraints.

I. INTRODUCTION

We expect autonomous general-purpose robots to navigate in unstructured, cluttered environments and perform multiple tasks autonomously. The robots need to guarantee the success of the task while responding reactively and safely to dynamic changes in the environment [1]–[6]. Robot motion generation encompasses all methods that define control commands to generate coordinated robot behaviors. These control commands take either the form of trajectory-level control [7]–[9] or direct, instantaneous controls such as joint velocity and acceleration [10]–[14].

On one side of the spectrum are motion planning approaches. Motion planning can be categorized into two subgroups based on the underlying methodology. First, sample-based methods provide probabilistic completeness guarantees in terms of goal-reaching and collision-free paths [8], [15]–[18]. Gradient-based trajectory optimization methods [7], [19], however, search for the optimal path to a task-specific goal, taking into account several underlying objectives, e.g., trajectory smoothness, obstacle avoidance, or joint limits avoidance. Algorithms from these two categories consider a static environment with a predefined and static goal location. These approaches are, thus, specifically suited for
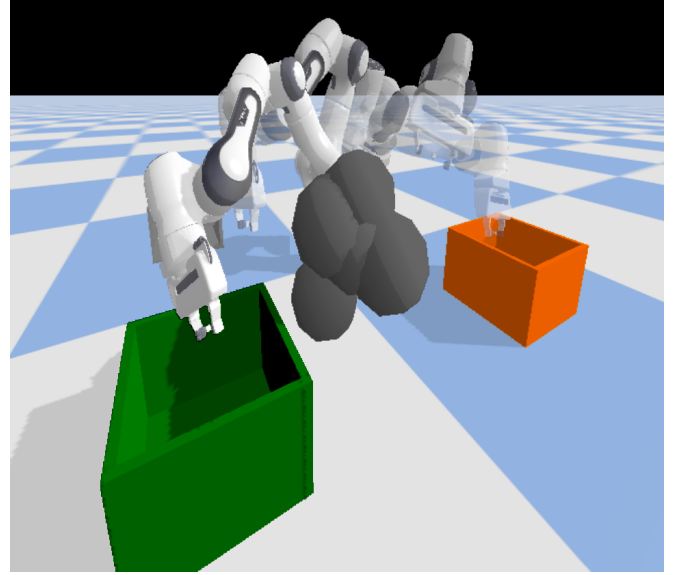
robot operations in structured environments. To enhance adaptability in dynamic environments, online planners [20]–[24], attempt to ensure reactivity through online re-planning over the action space for a short planning horizon. These methods find the optimal sequence of actions for a certain planning horizon. The first proposed action of the sequence is executed, followed by re-planning. While some efforts have been made to adapt these approaches to high-dimensional robots [22], these methods become computationally intensive in high dimensions, limiting the maximum frequency of control and, thus, reactivity.

Reactive motion generators are on the other side of the spectrum. These methods aim to meet high-frequency requirements to provide instantaneous control. Therefore, reactive motion generators ensure adaptability to environmental changes [25], [26] and hence local safety, e.g., by repulsive potential fields for obstacle avoidance. However, these methods do not provide look-ahead planning due to the high-frequency requirements. As a result, reactive motion generators are myopic. The myopic behavior makes the reactive methods susceptible to getting trapped in local minima when solving simultaneous objectives, such as obstacle and self-collision avoidance, as well as goal-reaching.

In this work, we propose a hierarchical policy blending as inference (HiPBI) approach that employs hierarchical decision making and probabilistic inference to combine the benefits of online motion planning and reactive motion

[1] Computer Science Department, Technische Universität Darmstadt (Germany), [2] German Research Center for AI (DFKI), [3] Hessian.AI, [4] Centre for Cognitive Science {`kay.hansel, julen.urain, jan.peters, georgia.chalvatzaki`} `@tu-darmstadt.de`

generators. On the lower level, we apply Riemannian motion policies (RMP) [6], [11]. RMPs represent myopic policies and provide a high-frequency response to environmental changes. On the higher level, we adopt planning-as-inference and use a sampling-based look-ahead planner that operates on the parameter space of the underlying RMPs. Therefore, we reformulate the RMPs as Gaussian policies to employ planning-as-inference methods [27], [28] resulting in a product of experts (PoE) [29], [30] formulation that belongs to the exponential family. To evaluate the performance of our approach, we conducted empirical studies for 2DoF point-mass navigation and 7DoF robot manipulation tasks in complex and dynamic environments. We compared our approach with representative reactive motion generation baselines. Our results highlight the effectiveness of HiPBI in solving tasks faced with cluttered and dynamic obstacles. By combining high-level planning with low-level reactive control, our approach achieves high success and safety rates.

Contribution: Our contributions are: (i) Developing a robot motion generation method that hierarchically combines reactive motion generation and sample-based online planning; (ii) Employing planning-as-inference to address the policy blending problem in an online fashion; (iii) Empirically demonstrating the achievement of feasible and reactive motions.

II. RELATED WORK

Hierarchical decision-making – abstracting the motion generation problem into multiple decision levels – is well-known practice in robotics. Such methods rely on multi-level planners or operate in the parameter space of motion policies. The former, such as TAMP, hierarchical planning, or hierarchical RL [31]–[34], generate subgoals that an underlying planner or policy must achieve. The latter either specifically adjusts constraint functions of dynamic motion primitives [35], [36] or selects a policy from a mixture of experts [37]–[42]. Given the multi-centric nature of robotic tasks, a hierarchical mixture of experts selects only one of the experts [40]. When faced with unexpected environmental changes, this selective behavior leads to suboptimal performance [38] or at least to a combination of experts with already complexly encoded behaviors, e.g., by imitation or reinforcement learning [40]. We argue that the composition of simple and stable reactive policies is capable of generating complex reactive behaviors in robotics [43]–[45].

The fundamental work for reactive motion generators applied artificial potential fields for modeling obstacle avoidance (repulsive) and goal-reaching (attractive) behavior [1]. This work formed the basis for operational space control investigating reactive policies to achieve instantaneous robot motion control [46], [47]. Using Riemannian metrics instead of Euclidean, the RMP framework extends operational space control considering geodesics near obstacles [6], [11], [44], [45]. The recently proposed geometric fabrics generalize RMPs by employing Finsler geometry rather than Riemannian one [12]–[14]. Motion primitives provide learning-based generation of reactive, stable behavior [4], [10], [48]. While most motion generators promise to be locally reactive, they are prone to get trapped in local minima.

RMP's intuitive superposition of policies corresponds to a product of experts [11], [45]. In the context of primitives, blending is useful to express complex behavior out of previously encoded primitives. Therefore, several works applied blending for parameterized motion primitives [5], [49], [50] or Gaussian processes [51]–[55]. Recent work addressed the blending problem utilizing QP optimization [56], energy-based models [9], [43], or learning from demonstrations [57]. However, most methods assume equal importance for all experts or adjust the importance offline through optimization or learning.

III. PRELIMINARIES

This section introduces the necessary background to frame the hierarchical model and stochastic optimization. First, we present a concept to express optimal control as a Bayesian inference problem. Second, we provide a mathematical object to describe the expert policies for reactive motion generation.

Control as Inference: Denoting the system state $s_t \in \mathbb{R}^s$ and action $a_t \in \mathbb{R}^a$ at time instant t, we define a discrete-time state-action trajectory as the sequence $\tau \triangleq (s_0, a_0, s_1, a_1, \ldots, s_T, a_T)$ over a time horizon T. Given the transition dynamics $p(s_{t+1} \mid s_t, a_t)$ and a policy $\pi(a_t \mid s_t; \theta)$ conditioned on the parameters θ, in optimal control, we aim to find the policy parameters θ that minimize the expected objective function $\mathcal{J}_\mathrm{C}(\tau)$, where $\theta^* = \arg\min_\theta \mathbb{E}_{p', \pi}[\mathcal{J}_\mathrm{C}(\tau; \theta)]$.

We express the objective function $\mathcal{J}_\mathrm{C}(\tau) = \sum_k c_k(\tau)$ as the sum of task-related cost functions, e.g., trajectory smoothness, collision avoidance, distance to target. Optimal control can be framed as a Bayesian inference problem considering the distribution over the policy parameters θ [28], [58], [59]. We introduce a binary random variable $\mathcal{O} \in \{0, 1\}$ that indicates the optimality of a trajectory τ w.r.t. the objective function $\mathcal{J}_\mathrm{C}(\tau)$. We express the probability of $\mathcal{O}=1$ given a trajectory τ by $p(\mathcal{O}=1|\tau) \propto \exp(-\mathcal{J}_\mathrm{C}(\tau))$. Given the trajectory distribution

$$p(\tau|\theta) = p(s_0) \prod_{t=0}^{T-1} p(s_{t+1}|s_t, a_t) \pi(a_t|s_t; \theta),$$

we can infer the optimality likelihood $p(\mathcal{O}=1|\theta)$ as the marginal probability over all the state and action trajectories,

$$p(\mathcal{O}=1|\theta) = \int_\tau p(\mathcal{O}=1|\tau) p(\tau|\theta) \, \mathrm{d}\tau.$$

Note that $\log p(\mathcal{O}=1|\theta) \propto -\mathcal{J}_\mathrm{C}(\tau; \theta)$. Given a prior distribution $p(\theta)$, we can approximate the posterior distribution given the optimality $\mathcal{O} = 1$ as $q(\theta|\mathcal{O}=1) \propto p(\mathcal{O}=1|\theta) p(\theta)$, that updates the distribution over θ towards the parameters that are optimal for our objective function $\mathcal{J}_\mathrm{C}(\tau)$.

Riemannian Motion Policies: RMPs [6], [11], [44] describe a mathematical object for representing reactive, modular motion generation policies. In RMPs, the action a_t at time instant t is computed by a weighted sum of a set of policy components, $a_t = \sum_i \alpha_i(s_t)\pi_i(s_t)$, with $\alpha_i > 0$ the weighting term for the component i. In RMPs, the state $s_t = (q_t, \dot{q}_t, c_t)$ represents the robot's position $q_t \in \mathbb{R}^q$,

velocity $\dot{q}_t \in \mathbb{R}^q$ and environment context c_t. The action is chosen to be the robot acceleration $a_t = \ddot{q}_t \in \mathbb{R}^q$. We assume a set of task maps $\phi : \mathcal{Q} \to \mathcal{X}$, that relate the robot configuration $\mathcal{Q}$ space and a certain task space $\mathcal{X}$. Then, given a task-space policy π_x, we can represent a policy in the robot configuration space by $\pi_q = \boldsymbol{J}_\phi^\dagger \pi_x(\phi(\boldsymbol{s}_t))$, with $\boldsymbol{J}_\phi^\dagger$ the Jacobian pseudoinverse of the task map ϕ.

IV. HIERARCHICAL REACTIVE POLICY BLENDING

We propose a hierarchical optimization framework to blend reactive policies for motion generation in complex and dynamic environments. Our approach, HiPBI, expresses the blended policy as a PoE. A weighted superposition of the hand-tuned reactive policies determines the optimal action at the low level. At the high level, we formalize the optimization of the weights as a probabilistic inference problem and employ a sampling-based look-ahead planner. The hierarchical optimization scheme offers (i) high-frequency control and, hence, fast adaptation to environmental changes; (ii) avoidance of local optima in cluttered and dense environments through the deliberate superposition of policies. In the following, we will discuss both sublevels sequentially.

Weighted superposition of reactive policies: We adopt a probabilistic viewpoint and formalize each policy component of the RMP as an energy-based model

$$\pi_i(\boldsymbol{a}_t \mid \boldsymbol{s}_t; \boldsymbol{\theta}_i) \propto \exp(-E_i(\boldsymbol{s}_t, \boldsymbol{a}_t; \boldsymbol{\theta}_i)),$$

taking the form of a Boltzmann distribution. The quantities $\boldsymbol{s}_t \in \mathcal{S}$ and $\boldsymbol{a}_t \in \mathcal{A}$ denote a state and action at time instance t, respectively. An energy function $E_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ assigns a cost to each state-action pair. Thus, the Boltzmann distribution gives a probability value to each state-action pair. The choice of the energy function E_i and its hyperparameter $\boldsymbol{\theta}_i$ is usually made in advance. In the case of Riemannian geometry, the energy $E_i(\boldsymbol{s}_t, \boldsymbol{a}_t; \boldsymbol{\theta}_i)$ is a quadratic function satisfying smoothness and convexity. Accordingly, the Boltzmann distribution forms a multivariate Gaussian $\pi_i(\boldsymbol{a}_t \mid \boldsymbol{s}_t; \boldsymbol{\theta}_i) = \mathcal{N}(\boldsymbol{\mu}_i(\boldsymbol{s}_t), \boldsymbol{\Lambda}_i(\boldsymbol{s}_t)^{-1})$ with $\boldsymbol{\mu}_i(\boldsymbol{s}_t)$ and $\boldsymbol{\Lambda}_i(\boldsymbol{s}_t)$ as the mean and the precision matrix, respectively [43]. Referring to RMPs, the mean is the forcing function, and the precision matrix's inverse corresponds to the Riemannian metric. We leverage a PoE

$$\pi(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{\beta}) \propto \prod_{i=1}^n \pi_i(\boldsymbol{a}_t \mid \boldsymbol{s}_t; \boldsymbol{\theta}_i)^{\beta_i}, \qquad (1)$$

with weighting factors $\boldsymbol{\beta}$, also known as *temperatures*, representing the importance or relevance of each policy in the product. In the logarithmic space, this blending equals a weighted superposition. The optimal action at time instance t results from $\boldsymbol{a}_t^* = \operatorname{argmax}_{\boldsymbol{a} \in \mathcal{A}} \log \pi(\boldsymbol{a} \mid \boldsymbol{s}_t, \boldsymbol{\beta})$, depending on state $\boldsymbol{s}_t$ and the weights $\boldsymbol{\beta}$. Due to the quadratic nature of the defined energy functions $E_i(\boldsymbol{s}_t, \boldsymbol{a}_t)$ from the RMP framework, the PoE is a Multivariate Gaussian distribution. Therefore, we can obtain the gradient and hence the optimal action analytically in closed form.

Sampling-based online planner: The behavior of the agent $\pi(\boldsymbol{s}_t, \boldsymbol{a}_t)$ and hence an applied action sequence $\boldsymbol{a}_t^*, \dots, \boldsymbol{a}_{t+h}^*$ up to time $t + h$ are induced by the superposition of n experts as in (1). The temperature values $\boldsymbol{\beta}$ give us the possibility to change the relevance or importance of an expert. In an online fashion, a change in the relevance of experts makes it possible to induce planning into the myopic nature of the policy $\pi(\boldsymbol{s}_t, \boldsymbol{a}_t)$. With the formulation of PoE in mind, we exploit the duality between control and probabilistic inference by formalizing an optimization procedure for deriving optimal weights. The posterior of optimal blending is given

$$p(\boldsymbol{\beta} \mid \mathcal{O}{=}1, \boldsymbol{s}_t) \propto p(\mathcal{O}{=}1 \mid \boldsymbol{\beta}, \boldsymbol{s}_t) \, p(\boldsymbol{\beta} \mid \boldsymbol{s}_t),$$

with the current state $\boldsymbol{s}_t$ and the optimal likelihood $p(\mathcal{O}{=}1 \mid \boldsymbol{\beta}, \boldsymbol{s}_t)$ [27], [28]. Given this likelihood, we can insert desired higher-level goals into the framework to achieve a desired behavior. Assuming a parameterized variational distribution $q(\boldsymbol{\beta}; \boldsymbol{\theta})$, we minimize the reverse Kullback–Leibler divergence

$$q^*(\boldsymbol{\beta}; \boldsymbol{\theta}) = \operatorname{argmin}_{q(\boldsymbol{\beta}; \boldsymbol{\theta})} \mathbb{D}_{\mathrm{KL}}[q(\boldsymbol{\beta}; \boldsymbol{\theta}) \parallel p(\boldsymbol{\beta} \mid \mathcal{O}{=}1, \boldsymbol{s}_t)].$$

The reverse KL divergence ensures that the optimization fits the distribution $q(\boldsymbol{\beta}; \boldsymbol{\theta})$ to modes of the $p(\boldsymbol{\beta} \mid \mathcal{O}{=}1, \boldsymbol{s}_t)$. Due to the fact that there are potentially many optimal solutions, the mode-seeking behavior is beneficial. The optimal distribution $q^*(\boldsymbol{\beta}; \boldsymbol{\theta})$ is therefore obtained from

$$q^*(\boldsymbol{\beta}; \boldsymbol{\theta}) = \min_{q(\boldsymbol{\beta}; \boldsymbol{\theta})} \left[\mathbb{H}_{q(\boldsymbol{\beta}, \boldsymbol{\theta})} \left[\frac{p(\mathcal{O}{=}1 \mid \boldsymbol{\beta}, \boldsymbol{s}_t) \, p(\boldsymbol{\beta} \mid \boldsymbol{s}_t)}{q(\boldsymbol{\beta}; \boldsymbol{\theta})} \right] \right].$$

Given the inference framework, we can impose important properties on the temperature parameters. Using an exponential distribution as $q(\boldsymbol{\beta}; \boldsymbol{\theta})$, we can ensure that all weights are greater than zero. Additionally, the use of a Dirichlet distribution implies that the temperatures sum to one. We choose $q(\boldsymbol{\beta}; \boldsymbol{\theta}) = \mathrm{Dir}(\boldsymbol{\beta}; \boldsymbol{\theta})$ for this reason. The prior $p(\boldsymbol{\beta} \mid \boldsymbol{s}_t)$ gives us the opportunity to incorporate prior knowledge into the framework. As we do not want to bias the optimization, we consider a uniform prior. The Shannon entropy of the variational distribution $q(\boldsymbol{\beta}; \boldsymbol{\theta})$ itself ensures that the distribution does not collapse. The optimal likelihood forms a marginal likelihood

$$p(\mathcal{O}{=}1 \mid \boldsymbol{\beta}, \boldsymbol{s}_t) = \int p(\mathcal{O}{=}1 \mid \boldsymbol{\tau}, \boldsymbol{s}_1) \, p(\boldsymbol{\tau} \mid \boldsymbol{\beta}, \boldsymbol{s}_1) \, \mathrm{d}\tau,$$

over possible trajectories $\boldsymbol{\tau}$. Inferring this quantity is challenging due to the integral over $\boldsymbol{\tau}$ on the right hand side. We employ variational inference that defines a lower bound, also known as evidence lower bound (ELBO) [60], [61]. Hence, we choose a variational distribution

$$\hat{q}(\boldsymbol{\tau} \mid \boldsymbol{\beta}) = \hat{q}(\boldsymbol{s}_t) \prod_{i=t}^h \hat{q}(\boldsymbol{s}_{t+1} \mid \boldsymbol{a}_t, \boldsymbol{s}_t) \pi(\boldsymbol{a}_t \mid \boldsymbol{\beta}, \boldsymbol{s}_t),$$

over $\boldsymbol{\tau}$. Assuming that $\hat{q}(\boldsymbol{\tau} \mid \boldsymbol{\beta})$ approximates the true trajectory distribution $p(\boldsymbol{\tau} \mid \boldsymbol{\beta}, \boldsymbol{s}_t)$ closely - given $\boldsymbol{\beta}$ and $\boldsymbol{s}_t$ - we obtain the ELBO

$$\mathbb{E}_{q(\boldsymbol{\beta})} \left[\mathbb{E}_{\hat{q}(\tau)} \left[\sum_{i=t}^h \log \frac{p(\mathcal{O}_t{=}1 \mid, \boldsymbol{a}_t, \boldsymbol{s}_t)}{\pi(\boldsymbol{a}_t \mid \boldsymbol{\beta}, \boldsymbol{s}_t)} \right] + \log q(\boldsymbol{\beta}) \right]. \quad (2)$$
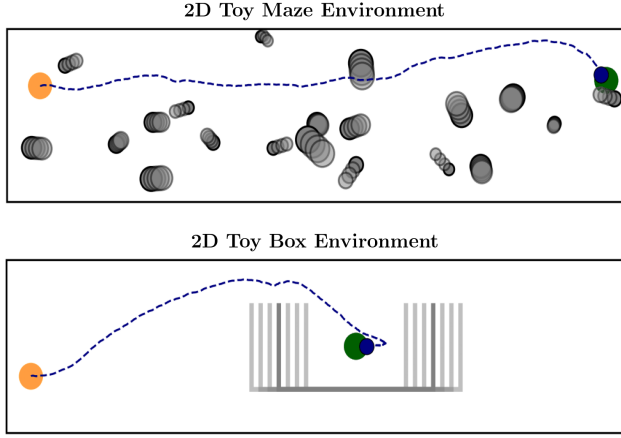
Fig. 2: 2D Toy environments for planar point-mass navigation. The orange dot denotes the start and the green one the goal location. **Top.** The toy maze environment with dynamic obstacles. **Bottom.** the toy box environment, in which a box moves horizontally at a constant speed. The goal is fixed in the center of the moving box.

For clarity, we omitted the dependencies on $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$ of $q(\boldsymbol{\beta} \mid \boldsymbol{\theta})$ and $\hat{q}(\tau \mid \boldsymbol{\beta})$. The first quantity in the final objective (2) resembles look-ahead planning. Thus, we approximate the optimal likelihood utilizing a shooting method. For the optimization of a parameterized distribution $q(\boldsymbol{\beta}; \boldsymbol{\theta})$ we can either apply gradient-based [62] or apply sampling-based techniques [21], [23], [23], [43]. Since online sampling-based techniques are promising in practice, we chose the iCEM approach [20], [21], i.e., after shooting, we take the k best samples and apply moment matching to them. As we can calculate the entropy $\mathbb{E}_{q(\boldsymbol{\beta})}\left[\log q(\boldsymbol{\beta})\right]$ in (2) of our parameterized distribution $q(\boldsymbol{\beta}; \boldsymbol{\theta})$ in closed form, it is a constant value applying iCEM. Hence, we end up with the final objective for the iCEM method results in

$$\mathcal{J}_{\text{CEM}}(\boldsymbol{\beta}) = \mathbb{E}_{q(\boldsymbol{\beta})\hat{q}(\tau)}\big[\textstyle\sum_{i=t}^{h} \log p(\mathcal{O}_t{=}1 \mid, \boldsymbol{a}_t, \boldsymbol{s}_t) \\ + \lambda_\pi \log \pi(\boldsymbol{a}_t \mid \boldsymbol{\beta}, \boldsymbol{s}_t) \big],$$

with regularization parameter λ_π. After selecting the k best samples, we estimate the mean and precision of the Dirichlet distribution separately [63]. For the precision, a Newton-Raphson like method is applied, while for the mean, a fixed-point iteration takes place.

V. EXPERIMENTS

In this section, we benchmark HiPBI against two baselines. The algorithm Riemannian motion policies (RMP*flow*) [11] works as a baseline from the family of reactive policies, that corresponds to a graph-based syntethis framework, and combines individual local RMPs to generate global dynamical behavior. A framework for real-time planning and second baseline is the improved cross entropy method for model-predictive-control (iCEM-MPC) [21] framework. The method utilizes improved cross entropy method (iCEM) for trajectory optimization in a model predictive control (MPC) scheme. First, two low-dimensional planar navigation environments give us insights into how the different algorithms behave under different environmental conditions. Then, we

consider a high-dimensional robotic simulation. This environment shows how HiPBI adapts to high-dimensional state and action spaces. In all experiments, we set $\lambda_\pi = 0$.

Toy Environments: The 2D toy maze environment (2D-Maze) is a dynamic 2D planar environment on which a particle navigates from a random start to a random goal position (see Fig. 2). In the environment, a given number of m circular obstacles – partly static, partly dynamic – bar the way. The environment randomly sets obstacles inside a restricted area between the start (orange point) and goal positions (green point). We model the movement of the obstacles using a constant velocity model. 2D-Maze mimics a dense, cluttered and dynamic environment.

Unlike 2D-Maze, the 2D toy box environment (2D-Box) presents a dynamic domain in which constant local optima exist (see Fig. 2). The box is dynamic, and its motion is modeled as a constant velocity one. The start position is sampled randomly to the right or left of the box. The challenge comes from local optima under the box or in front, i.e., to the left or the right. Furthermore, the dynamic nature complicates the planning of a feasible solution. Validation is important as local optima are constantly changing in 2D-Maze – they appear and disappear independently - whereas they exist permanently in 2D-Box. Therefore, 2D-Box shows us the effectiveness in overcoming constant local optima.

We consider four different metrics for validation in 2D-Maze and 2D-Box: (i) the success rate (SUC), indicating the percentage of times the goal has been reached; (ii) the safety rate (SAFE), implying collision-free motions; (iii) the final l2 distance (L2D) to the goal; and (iv) the needed time steps (TS) until the goal is reached. As RMP*flow* runs at high frequency, we studied iCEM-MPC and HiPBI in a synchronous (S) and asynchronous (A) mode. In the former, iCEM-MPC and HiPBI have sufficient time to find feasible solutions – as the environment remains fixed during the planning. In the latter, planning runs asynchronously with the environment, and online planning is needed. Therefore, we applied different look ahead (LA) horizons for both the basic iCEM-MPC and the HiPBI methods.

RMP*flow* employs the composition of experts with attractive and repulsive forces. To achieve a curling behavior, we choose an expert $\pi_{\text{curl}} \perp \pi_{\text{goal}}$ that exerts forces normal to the goal attractive force. π_{curl} scales proportional to π_{goal} and, thus, vanishes if the particle reaches the goal. We apply two mutually balancing agents to avoid constant rotational forces, i.e., $\pi_{\text{curl}_i} = -\pi_{\text{curl}_j}$. Although this extension does not affect RMP*flow*, HiPBI adapts the weights and hence achieves curling behavior.

Table I summarizes our results. In 2D-Box, RMP*flow* converges to local optima. In synchronous mode, iCEM-MPC and HiPBI improve their success rate with increasing look-ahead size and encounter no collisions. However, iCEM-MPC loses performance in asynchronous mode as it cannot react fast to environmental changes. HiPBI, which combines online planning with reactive control, achieves a 100% success and safety rate with a look-ahead of 75 without suffering from the same performance gap. In 2D-

Success Rate on 2D Toy Box Environment

	0	5	10	15	20	25	30
RMP*flow*	0.0	0.0	0.0	0.0	19.0	30.0	32.0
iCEM (LA 25)	31.0	33.3	38.0	27.0	45.0	39.4	38.0
iCEM (LA 50)	52.0	42.0	56.0	49.0	54.0	62.0	64.0
iCEM (LA 75)	78.0	77.0	76.0	76.0	76.0	67.0	70.0
iCEM (LA 100)	58.0	48.0	56.0	52.0	46.0	48.0	38.0
HiPBI (LA 25)	4.0	2.0	3.0	4.0	30.0	30.0	25.0
HiPBI (LA 50)	72.0	69.0	80.0	79.0	89.0	89.0	58.0
HiPBI (LA 75)	100.0	100.0	100.0	99.0	94.0	96.0	76.0
HiPBI (LA 100)	100.0	100.0	100.0	100.0	89.0	85.0	77.0

Different speed Levels

Safety Rate on 2D Toy Box Environment

	0	5	10	15	20	25	30
	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	31.0	33.3	38.0	27.0	45.0	39.4	38.0
	59.0	52.0	64.0	58.0	68.0	68.0	72.0
	84.0	81.0	80.0	79.0	81.0	70.0	74.0
	66.0	64.0	71.0	67.0	62.0	68.0	57.0
	100.0	100.0	100.0	100.0	99.0	98.0	92.0
	100.0	100.0	100.0	99.0	100.0	99.0	95.0
	100.0	100.0	100.0	99.0	99.0	98.0	94.0
	100.0	100.0	100.0	100.0	91.0	86.0	84.0

Different Speed Levels

Fig. 3: The results of an ablation study in the 2D toy box environment. The speed increases from a minimum of zero (static) to 30 pixels per step (dynamic). We compare the baselines, i.e., RMP*flow* and iCEM-MPC, with our method HiPBI and employ different look-ahead horizons (LA). **Left.** The success rate shows the performance. **Right.** The safety rate indicates how often no collision occurs.

Maze, RMP*flow* achieves a success rate of 77%. As in 2D-Box, iCEM-MPC and HiPBI perform well in synchronous mode. However, in the asynchronous mode, we notice that iCEM-MPC suffers again from its slow response to environmental changes, more noticeable at higher look-ahead values. Interestingly, this behavior is reversed in 2D-Box. Dodging under or over one box is simpler than avoiding multiple particles. The performance of HiPBI improves with increasing look-ahead planning. The similar collision rate as RMP*flow* is reasonable as both use the same parameters for the underlying RMP. Due to planning, HiPBI improves the success rate compared to the myopic RMP*flow*.

In Figure 3, we show a comparison on 2D-Box with different speed levels of the box. This comparison provides insights into the responsiveness to environmental changes. We see how success and safety rates change as velocity increases. Regardless of the velocity, RMP*flow* does not collide with the box environment but has a low success rate. iCEM-MPC achieves a higher success rate, but collisions occur more often. The ablation study highlights how HiPBI combines the advantages of low-level reactiveness and high-level planning. Increasing speed has a small influence on HiPBI, with only a slight drop in performance at a speed level of 30 pixels per step while maintaining a sufficient safety rate.

Manipulation Environment: We investigate the performance of HiPBI on a high-dimensional robotics task with a 7DoF manipulator in the physics engine PyBullet [64]. Fig. 5 shows the arm surrounded by four boxes. In each round, the robot must first get to a randomly selected intermediate goal (orange box). After it reaches the intermediate goal, it has to reach the final goal (green box). Several sphere-like objects block the way during the path from the intermediate to the goal box. Thus, this task resembles a high-dimensional pick-and-place task involving multiple local optima.

In an ablation study, Fig. 4, we compared the performance of HiPBI against RMP*flow*. Long horizon planning was not feasible with iCEM-MPC, thus we do not add any comparison, as the short horizons did not give satisfactory results. We consider two modes, a static one and a dynamic one. In the former, zero to five static spheres are randomly sampled in a predefined space between the intermediate and goal box. In the latter, we use one to five dynamic spheres similar to the former case assuming constant velocity models. The dynamic obstacles are restricted to staying within the path of the panda and goal box. RMP*flow* uses eight experts, which included self-collision avoidance, joint and velocity limitations, goal-reaching, and avoidance of obstacles such as floors, boxes, and spheres. Unlike in 2D-Box and 2D-Maze, HiPBI leverages the same experts and omits the local

TABLE I: Evaluation of the baselines, i.e., RMP*flow* and iCEM-MPC, and our method HiPBI on the planar point-mass navigation tasks using different metrics: (i) the success rate (SUC); (ii) the safety rate (SAFE); (iii) the L2 distance from the final state to the goal (L2D); and (iv) the time steps required to reach the goal (TS). We employed different look-ahead horizons depicted as LA. The quantities S and A indicate whether the dynamics run synchronously with the algorithm or asynchronously. **Left.** Experiments took place in the 2D toy box environment. **Right.** Results in the 2D toy maze environment are highlighted.

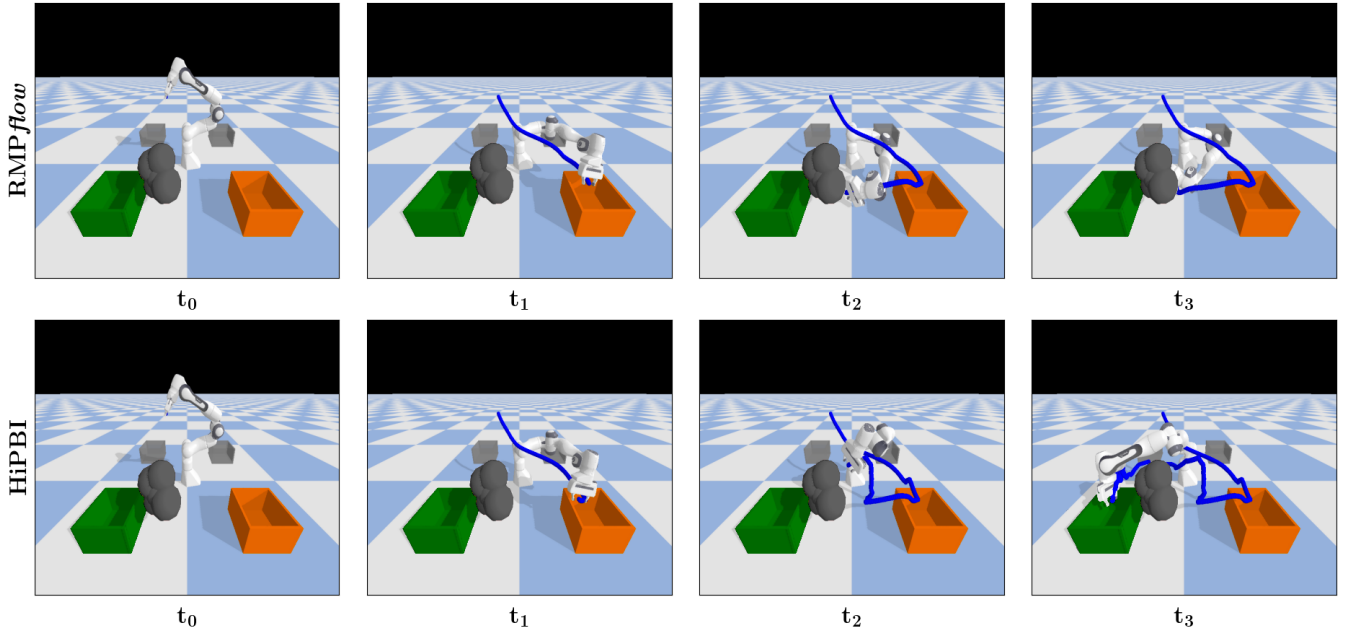|	2D Toy Box Environment				2D Toy Maze Environment			
	SUC[%]	SAFE[%]	L2D	TS	SUC[%]	SAFE[%]	L2D	TS
RMP*flow*	0	**100**	198.9 ± 1.5	500.0 ± 0.0	77	89	161.5 ± 620.0	330.7 ± 191.3
iCEM (LA 25, S)	38	**100**	77.2 ± 91.2	353.9 ± 277.9	98	**99**	$\mathbf{17.4 \pm 175.1}$	$\mathbf{133.1 \pm 65.4}$
iCEM (LA 50, S)	57	**100**	56.6 ± 73.6	271.0 ± 282.5	**99**	99	21.1 ± 163.1	167.0 ± 59.8
iCEM (LA 75, S)	90	**100**	43.3 ± 82.7	195.2 ± 177.2	**99**	99	37.7 ± 166.7	223.8 ± 90.5
HiPBI (LA 25, S)	2	**100**	189.3 ± 44.7	490.9 ± 81.8	98	99	20.3 ± 172.7	247.6 ± 55.8
HiPBI (LA 50, S)	61	**100**	49.5 ± 75.6	276.6 ± 251.2	**99**	99	17.5 ± 162.6	247.5 ± 47.6
HiPBI (LA 75, S)	**100**	**100**	7.3 ± 5.9	$\mathbf{131.9 \pm 18.0}$	**99**	99	19.0 ± 171.7	252.1 ± 47.3
iCEM (LA 25, A)	31	31	95.5 ± 94.8	372.9 ± 265.6	40	40	409.6 ± 570.0	356.6 ± 245.5
iCEM (LA 50, A)	54	64	64.3 ± 83.2	292.5 ± 271.7	4	4	774.0 ± 436.1	488.9 ± 78.1
iCEM (LA 75, A)	79	85	63.1 ± 84.4	237.7 ± 208.6	0	0	974.1 ± 287.5	499.2 ± 22.4
HiPBI (LA 25, A)	7	**100**	178.6 ± 71.1	477.1 ± 120.3	83	84	116.2 ± 386.3	294.2 ± 131.4
HiPBI (LA 50, A)	73	**100**	40.1 ± 76.9	324.3 ± 169.7	85	**87**	$\mathbf{100.0 \pm 357.9}$	$\mathbf{293.4 \pm 123.7}$
HiPBI (LA 75, A)	**100**	**100**	$\mathbf{8.5 \pm 6.0}$	$\mathbf{205.8 \pm 35.3}$	**86**	**87**	106.1 ± 376.5	297.3 ± 122.1

Fig. 4: Manipulation environment in which the intermediate (orange) and target (green) boxes are randomly selected out of four boxes. Five randomly generated grey obstacles obstruct the path of the 7DoF manipulator robot. With blue, we denote the executed trajectory. **Top.** Performance of Riemannian motion policies (RMP*flow*) method that gets stuck in a local optimum. **Bottom.** Performance of our proposed HiPBI, that successfully discovers an obstacle-free path to the target.

curling policies. Our approach adjusts the importance of the experts to achieve desired dynamical behavior. Due to the computational complexity of planning algorithms and the advantage of optimizing in parameter space, we apply the HiPBI in asynchronous mode – as 2D results confirm our assumption. While the high-level planner optimizes at a lower frequency, the local policies ensure reactive behavior. By choosing a Dirichlet distribution, we guarantee that each expert affects the dynamic system. Thus, no scenarios arise in that local policies are switched off.

In Figure 5, we present an ablation study and see that RMP*flow* performs better in a dynamic setting, strengthening our assumption that local optima alter in dynamic environments. However, the success rate decreases with an increasing number of obstacles. This outcome is reasonable, as each obstacle induces another constraint creating more local optima. HiPBI demonstrates significantly improved results with a look-ahead horizon of 25, corresponding to $2.5\,s$ at a

planning frequency of $10\,Hz$. In the static environment with fixed local optima, HiPBI also outperforms RMP*flow*.

Figure 4 compares two executed trajectories of RMP*flow* (top) and HiPBI (bottom) at four different time points. RMP*flow* naturally follows the myopic behavior and ends in a local optimum. HiPBI, on the other hand, exploits the information of the hierarchical high-level planning scheme. The reactive leaping motion indicates the low-level RMPs. High-level planning enables feasible solutions avoiding local optima and reaching the goal.

VI. CONCLUSIONS

We presented hierarchical policy blending as inference (HiPBI), a method for reactive motion generation that combines, at the low level, myopic reactive motion policies that can be modeled as a product of experts (PoE), and, on the high level, a sampling-based online planner on the parameter space of the policies, that decides over the optimal weighting of the experts. Our method dynamically adapts the importance of the different policies and shows superior performance in terms of task success-rate and safety (in terms of collision avoidance) against representative baselines, as demonstrated both in complex planar environments, and in high-dimensional robotic manipulation tasks in face of clutter and dynamic changes.

As our method comes with the cost of higher computational complexity, we will explore collocation methods for the planning process in the future and apply the approach to appropriately designed real robot environments. Furthermore, the probabilistic inference framework assumes a prior distribution to provide prior knowledge to the system. We will discuss the form and realization of such a prior, e.g., by imitation or offline reinforcement learning.

	Static Environment						Dynamic Environment				
RMP*flow*	78.4	43.1	20.0	23.5	21.6	25.5	90.2	76.5	82.4	64.7	64.7
HiPBI (LA 25)	100.0	64.0	58.2	59.5	61.4	58.2	94.1	93.5	85.6	81.7	86.3
HiPBI (LA 50)	100.0	72.3	66.7	61.9	63.2	70.6	96.7	90.2	90.9	88.3	83.6
HiPBI (LA 75)	98.0	71.2	64.1	65.4	62.5	58.2	96.1	89.5	90.2	88.3	79.1
	0	1	2	3	4	5	1	2	3	4	5
	Number of Obstacles						Number of Obstacles				

Fig. 5: Evaluation study on the manipulation environment. We benchmark our approach HiPBI in a static and a dynamic setting against the baseline RMP*flow*. **Left.** The success rate in a static environment. The number of obstacles varies from zero up to a maximum of five. **Right.** The success rate in a dynamic environment. A maximum of five movable obstacles are used.

REFERENCES

[1] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[2] D.-H. Park, H. Hoffmann, P. Pastor, and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2008, pp. 91–98.

[3] N. Hogan and D. Sternad, "Dynamic primitives of motor behavior," *Biological cybernetics*, vol. 106, no. 11, pp. 727–739, 2012.

[4] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.

[5] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Using probabilistic movement primitives in robotics," *Autonomous Robots*, vol. 42, no. 3, pp. 529–551, 2018.

[6] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian motion policies," *arXiv preprint arXiv:1801.02854*, 2018.

[7] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[8] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4569–4574.

[9] A. Lambert, A. T. Le, J. Urain, G. Chalvatzaki, B. Boots, and J. Peters, "Learning implicit priors for motion optimization," *arXiv preprint arXiv:2204.05369*, 2022.

[10] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[11] C.-A. Cheng, M. Mukadam, J. Issac, S. Birchfield, D. Fox, B. Boots, and N. Ratliff, "Rmpflow: A computational graph for automatic motion policy generation," in *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2018, pp. 441–457.

[12] M. Xie, K. Van Wyk, A. Li, M. A. Rana, Q. Wan, D. Fox, B. Boots, and N. Ratliff, "Geometric fabrics for the acceleration-based design of robotic motion," *arXiv preprint arXiv:2010.14750*, 2020.

[13] N. D. Ratliff, K. Van Wyk, M. Xie, A. Li, and M. A. Rana, "Generalized nonlinear and finsler geometry for robotics," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 206–10 212.

[14] K. Van Wyk, M. Xie, A. Li, M. A. Rana, B. Babich, B. Peele, Q. Wan, I. Akinola, B. Sundaralingam, D. Fox, *et al.*, "Geometric fabrics: Generalizing classical mechanics to capture the physics of behavior," *IEEE Robotics and Automation Letters*, 2022.

[15] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[16] S. LAVALLE, "Rapidly-exploring random trees: A new tool for path planning," *Computer Science Dept. Oct.*, vol. 98, no. 11, 1998.

[17] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[18] T. Löw, T. Bandyopadhyay, J. Williams, and P. V. Borges, "Prompt: Probabilistic motion primitives based trajectory planning." in *Robotics: Science and Systems*, 2021.

[19] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time gaussian process motion planning via probabilistic inference," *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1319–1340, 2018.

[20] M. Kobilarov, "Cross-entropy motion planning," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 855–871, 2012.

[21] 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 *Proceedings of the 2020 Conference on Robot Learning*, vol. 155. PMLR, 2021, pp. 1049–1065.

[22] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. D. Ratliff, D. Fox, F. Ramos, and B. Boots, "Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation," in *Proceedings of the 5th Conference on Robot Learning*, vol. 164. PMLR, 2022, pp. 750–759.

[23] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1714–1721.

[24] A. Lambert, A. Fishman, D. Fox, B. Boots, and F. Ramos, "Stein variational model predictive control," *arXiv preprint arXiv:2011.07641*, 2020.

[25] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous robot vehicles*. Springer, 1986, pp. 396–404.

[26] S. Calinon, "Robot learning with task-parameterized generative models," in *Robotics Research*. Springer, 2018, pp. 111–126.

[27] M. Toussaint, "Robot trajectory optimization using approximate inference," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 1049–1056.

[28] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," *arXiv preprint arXiv:1805.00909*, 2018.

[29] V. Tresp, "A Bayesian Committee Machine," *Neural Computation*, vol. 12, no. 11, pp. 2719–2741, 2000.

[30] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[31] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical planning in the now," in *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[32] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 639–646.

[33] S. Jauhri, J. Peters, and G. Chalvatzaki, "Robot learning of mobile manipulation with reachability behavior priors," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8399–8406, 2022.

[34] K. Pertsch, O. Rybkin, F. Ebert, S. Zhou, D. Jayaraman, C. Finn, and S. Levine, "Long-horizon visual planning with goal-conditioned hierarchical predictors," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 321–17 333, 2020.

[35] S. Bahl, M. Mukadam, A. Gupta, and D. Pathak, "Neural dynamic policies for end-to-end sensorimotor learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5058–5069, 2020.

[36] S. Bahl, A. Gupta, and D. Pathak, "Hierarchical neural dynamic policies," in *Robotics: Science and Systems*, 2021.

[37] C. Daniel, G. Neumann, and J. Peters, "Hierarchical relative entropy policy search," in *Artificial Intelligence and Statistics*. PMLR, 2012, pp. 273–281.

[38] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, and J. Peters, "Towards learning hierarchical skills for multi-phase manipulation tasks," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 1503–1510.

[39] F. End, R. Akrour, J. Peters, and G. Neumann, "Layered direct policy search for learning hierarchical skills," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 6442–6448.

[40] O. Celik, D. Zhou, G. Li, P. Becker, and G. Neumann, "Specializing versatile skill libraries using local mixture of experts," in *Conference on Robot Learning*. PMLR, 2022, pp. 1423–1433.

[41] R. Akrour, D. Tateo, and J. Peters, "Continuous action reinforcement learning from a mixture of interpretable experts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[42] M. Zaki, A. Mohan, A. Gopalan, and S. Mannor, "Actor-critic based improper reinforcement learning," in *Proceedings of the 39th International Conference on Machine Learning*, vol. 162. PMLR, 2022, pp. 25 867–25 919.

[43] J. Urain, A. Li, P. Liu, C. D'Eramo, and J. Peters, "Composable energy policies for reactive motion generation and reinforcement learning," *arXiv preprint arXiv:2105.04962*, 2021.

[44] A. Li, C.-A. Cheng, M. A. Rana, M. Xie, K. Van Wyk, N. Ratliff, and B. Boots, "Rmp2: A structured composable policy class for robot learning," *arXiv preprint arXiv:2103.05922*, 2021.

[45] M. Mukadam, C.-A. Cheng, D. Fox, B. Boots, and N. Ratliff, "Riemannian motion policy fusion through learnable lyapunov function reshaping," in *Conference on robot learning*. PMLR, 2020, pp. 204–219.

[46] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: A theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, 2008.

[47] M. Toussaint and C. Goerick, "A bayesian view on motor control and planning." *From Motor Learning to Interaction Learning in Robots*, vol. 264, pp. 227–252, 2010.

[48] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3339–3344.

[49] T. Luksch, M. Gienger, M. Mühlig, and T. Yoshiike, "Adaptive movement sequences and predictive decisions based on hierarchical dynamical systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2082–2088.

[50] M. Saveriano, F. Franzel, and D. Lee, "Merging position and orientation motion primitives," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7041–7047.

[51] Y. Cao and D. J. Fleet, "Generalized product of experts for automatic and principled fusion of gaussian process predictions," *arXiv preprint arXiv:1410.7827*, 2014.

[52] ——, "Transductive log opinion pool of gaussian process experts," *arXiv preprint arXiv:1511.07551*, 2015.

[53] M. Deisenroth and J. W. Ng, "Distributed gaussian processes," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37. PMLR, 2015, pp. 1481–1490.

[54] H. Liu, J. Cai, Y. Wang, and Y. S. Ong, "Generalized robust Bayesian committee machine for large-scale Gaussian process regression," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80. PMLR, 2018, pp. 3131–3140.

[55] S. Cohen, R. Mbuvha, T. Marwala, and M. Deisenroth, "Healing products of Gaussian process experts," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119. PMLR, 2020, pp. 2068–2077.

[56] N. Jaquier, Y. Zhou, J. Starke, and T. Asfour, "Learning to sequence and blend robot skills via differentiable optimization," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8431–8438, 2022.

[57] E. Pignat, J. Silvério, and S. Calinon, "Learning from demonstration using products of experts: Applications to manipulation and task prioritization," *The International Journal of Robotics Research*, vol. 41, no. 2, pp. 163–188, 2022.

[58] M. Botvinick and M. Toussaint, "Planning as inference," *Trends in cognitive sciences*, vol. 16, no. 10, pp. 485–488, 2012.

[59] K. Rawlik, M. Toussaint, and S. Vijayakumar, "On stochastic optimal control and reinforcement learning by approximate inference," *Proceedings of Robotics: Science and Systems VIII*, 2012.

[60] M. J. Beal, *Variational algorithms for approximate Bayesian inference*. University of London, University College London (United Kingdom), 2003.

[61] D. G. Tzikas, A. C. Likas, and N. P. Galatsanos, "The variational approximation for bayesian inference," *IEEE Signal Processing Magazine*, vol. 25, no. 6, pp. 131–146, 2008.

[62] J. Peters, K. Mulling, and Y. Altun, "Relative entropy policy search," in *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[63] T. Minka, "Estimating a dirichlet distribution," 2000.

[64] B. Ellenberger, "Pybullet gymperium," https://github.com/benelot/pybullet-gym, 2018–2019.