Compliant skills acquisition and multi-optima policy search with EM-based reinforcement learning

Sylvain Calinon, Petar Kormushev and Darwin G. Caldwell*

Department of Advanced Robotics, Istituto Italiano di Tecnologia (IIT), Genova, Italy.

Abstract

The democratization of robotics technology and the development of new actuators progressively bring robots closer to humans. The applications that can now be envisaged drastically contrast with the requirements of industrial robots. In standard manufacturing settings, the criterions used to assess performance are usually related to the robot's accuracy, repeatability, speed or stiffness. Learning a control policy to actuate such robots is characterized by the search of a single solution for the task, with a representation of the policy consisting of moving the robot through a set of points to follow a trajectory. With new environments such as homes and offices populated with humans, the reproduction performance is portrayed differently. These robots are expected to acquire rich motor skills that can be generalized to new situations, while behaving safely in the vicinity of users. Skills acquisition can no longer be guided by a single form of learning, and must instead combine different approaches to continuously create, adapt and refine policies. The family of search strategies based on expectation-maximization (EM) looks particularly promising to cope with these new requirements. The exploration can be performed directly in the policy parameters space, by refining the policy together with exploration parameters represented in the form of covariances. With this formulation, RL can be extended to a multi-optima search problem in which several policy alternatives can be considered. We present here two applications exploiting EM-based exploration strategies, by considering parameterized policies based on dynamical systems, and by using Gaussian mixture models for the search of multiple policy alternatives.

Keywords: reinforcement learning, learning by imitation, skills transfer, expectation-maximization, dynamical systems, Gaussian mixture model

1. Introduction

Skills acquisition encompasses various forms of learning. While some tasks can most successfully be transferred by *programming by demonstration* (PbD) [1, 2, 3], other tasks can more effectively be acquired by *reinforcement learning* (RL) [4, 5, 6]. Often, the efficiency lies in the interconnections of these imitation and self-improvement strategies. As with humans, a robot

*This work was partially supported by the Stiff-Flop European project under contract FP7-ICT-287728, and by the Pandora European project under contract FP7-ICT-288273. Authors email: `name.surname@iit.it`

should be able to acquire new skills by employing both mechanisms. Further, with new developments in robot actuation and human-robot interaction, it is increasingly important to bring these multiple strategies into play at different phases of interaction and practice. To make them work efficiently together, the design of compact and versatile representations of control policies becomes a crucial element for the overall success of the skill transfer process.

Reinforcement learning (RL) problems are often studied in the form of a Markov decision process (MDP) with the policy describing a state-action mapping where the states or/and actions can take a discrete form. An alternative view of the problem is to consider a direct policy search strategy where the policy is represented by a set of parameters that are stochastically sampled during exploration [7]. In the case of episodic rewards, this view of the problem shares similarities with other search procedures in stochastic optimization and evolutionary computation such as the *cross-entropy method* (CEM) [8] or the *covariance matrix adaptation evolution strategy* (CMA-ES) [9].

A tremendous effort within the machine learning and robotics community has been directed to moving RL to continuous real-world domains. Policy gradient methods have for example been proposed to cope with continuous states and actions, thus extending the potential for robotic applications. A policy gradient RL algorithm follows at each iteration the stochastic gradient of the policy performance until convergence at a local maximum. In order to devise robust ways of estimating the gradient, policy gradient algorithms such as *episodic REINFORCE* [10] or *episodic natural actor-critic* (eNAC) [4] have been developed to cope with high dimensionality continuous problems. Such algorithms have demonstrated very successful applications in robotics, but one shortcoming is that they depend on a learning rate that can sometimes be difficult to set, and that is essential for good performance [7, 11].

Such gradient methods converge to the closest local optimum. In unstructured environments, the search of a single policy can limit the robot's capability to rapidly cope with unpredictable events or new situations. It thus looks interesting to explore in parallel for possibly multiple policy options, such that the robot can exploit alternative solutions on-the-fly without having to find a new local optimum (e.g., when the previously best policy is not available anymore). Namely, by considering various optional sets of policy parameters producing the same (or similar) results, without having to represent those as distinct behaviors. Similarly, instead of considering only a single point in the policy parameters space producing the highest reward, it looks interesting to represent the region around this point in which high reward values can be obtained. This would bring additional information about how the robot can depart from a desired set of policy parameters while still reproducing the skill correctly.

A recent alternative view of RL problems proposes to eliminate gradient computation by transforming it into a probabilistic estimation approach [12, 13]. Such an approach appears to be a good candidate to address the above questions. The idea is to treat rewards as probabilities, which can be easily achieved by transforming the reward function adequately (e.g., by defining it with an exponential form). An *expectation-maximization* (EM) procedure can then be used to iteratively optimize the policy parameters, where the actions are treated as unobserved variables and the rewards are treated as probability distributions. The core of the approach shares similarities with CEM [8] and CMA-ES [9]. We will show in this article examples of application where this representation can be exploited to study the new challenges discussed above. This formulation opens new roads to further developments for which probabilistic approaches in machine learning can be exploited in robot learning by imitation and exploration.

The article is organized as follows. Section 2 presents EM-based reinforcement learning. Section 3 shows how the approach can be used to learn policies encapsulating motion and compliance information through a dynamical systems parameterization. An example is given with a pancake flipping task learned by a 7 DOFs robot manipulator (Section 3.1). Section 4 discusses how covariance information can be exploited in the search process, by focussing on multi-optima policy search (Section 4.1). An example is given with a skittle game (Section 4.2). Conclusions and future work are presented in Section 5.

2. EM-based reinforcement learning

Dayan and Hinton originally suggested that a RL problem can be tackled by EM [14]. They introduced the core idea of treating immediate rewards as probabilities of a fictitious event, in which case probabilistic inference techniques like EM can be used for optimization. They showed that in some circumstances, it is possible to make large well-founded changes to the policy parameters without explicitly estimating the curvature of the space of expected payoffs, by a mapping onto a maximum likelihood probability density estimation problem. In effect, they maximize the reward by solving a sequence of probability matching problems, where the policy parameters are chosen at each step to match a fictitious distribution determined by the average rewards experienced on the previous steps. Although there can be large changes in the policy parameters from one step to the next, there is a guarantee that the average reward monotonically increases.

From this simple idea, a growing interest in EM-based RL has emerged in the robotics community [13, 11, 5, 6], due to the flexible exploration-exploitation properties that the approach offers, and to the possibility of employing it with various forms of policy parameterization.

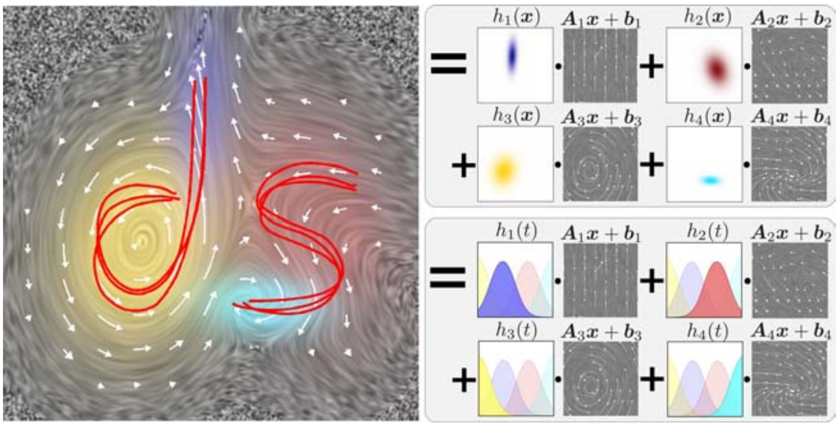
Policy improvement with path integrals (PI^2) [5, 6] consists of transforming a stochastic optimal control problem into the approximation of a path integral. It is derived by transforming the optimal control problem from a constrained minimization to a maximum likelihood formulation, avoiding the computation of a gradient. The approach takes inspiration from recent work showing that for a class of discrete stochastic optimal control problems, the Bellman equation can be written as the Kullback-Leibler divergence between the probability distribution of the controlled and uncontrolled dynamics [15].

In *policy learning by weighting exploration with the returns* (PoWER) [13], the action is treated as an unobserved variable and the returns are considered as a probability distribution. It relies on the following intuition: a safe way to generate new policies is to look in the convex combination of sampled policies. The algorithm searches for ways of improving the current best policy by staying close to the explored policies with high returns and far from solutions with low returns.

PoWER estimates a policy $\mathbf{\Theta}^{(n)}$ (at iteration n) such as to maximize the lower bound on the expected return from following the policy. As a simple form of importance sampling, an ordered set of the best policies $\{\mathbf{\Theta}_k\}_{k=1}^{M}$ obtained so far can be used, with $r(\mathbf{\Theta}_1) \geq r(\mathbf{\Theta}_2) \geq \ldots \geq r(\mathbf{\Theta}_M)$. In the simplest form of the algorithm, a new set of policy parameters is estimated and added to the training set with

$$\mathbf{\Theta}^{(n)} = \mathbf{\Theta}^{(n-1)} + \frac{\sum_{m}^{M} r(\mathbf{\Theta}_m)\Big[\mathbf{\Theta}_m - \mathbf{\Theta}^{(n-1)}\Big]}{\sum_{m}^{M} r(\mathbf{\Theta}_m)}. \quad (1)$$

The above equation is written in a form that emphasizes the correspondences/differences with gradient-based approaches. $\Big[\mathbf{\Theta}_m - \mathbf{\Theta}^{(n-1)}\Big]$ represents the relative exploration between the policy



=
$h_1(\boldsymbol{x})$
$\boldsymbol{A}_1\boldsymbol{x}+\boldsymbol{b}_1$
+
$h_2(\boldsymbol{x})$
$\boldsymbol{A}_2\boldsymbol{x}+\boldsymbol{b}_2$
+
$h_3(\boldsymbol{x})$
$\boldsymbol{A}_3\boldsymbol{x}+\boldsymbol{b}_3$
+
$h_4(\boldsymbol{x})$
$\boldsymbol{A}_4\boldsymbol{x}+\boldsymbol{b}_4$
=
$h_1(t)$
$\boldsymbol{A}_1\boldsymbol{x}+\boldsymbol{b}_1$
+
$h_2(t)$
$\boldsymbol{A}_2\boldsymbol{x}+\boldsymbol{b}_2$
+
$h_3(t)$
$\boldsymbol{A}_3\boldsymbol{x}+\boldsymbol{b}_3$
+
$h_4(t)$
$\boldsymbol{A}_4\boldsymbol{x}+\boldsymbol{b}_4$

Figure 1: Illustration of a compact encoding of movement as a weighted sum of linear systems.

parameters used in the *m*-th ordered trial and the current best policy parameters. This difference is weighted by the corresponding return $r(\boldsymbol{\Theta}_m)$, and normalized. This equation can also be rewritten as a weighted sum of the best policies obtained so far, thus making links with imitation learning strategies where each iteration in the process corresponds to the imitation of the most successful policies. This last form emphasizes the links with state-of-the-art algorithms in stochastic optimization and evolutionary computation such as the *cross-entropy method* (CEM) [8] and the *covariance matrix adaptation evolution strategy* (CMA-ES) [9].

In the next section, we exploit this EM-based RL mechanism in a compliant robot learning to flip artificial pancakes.

3. Learning of trajectory and compliance information with dynamical systems

A widely adopted approach to the representation of complex skills is to decompose nonlinear movements into smaller units of actions. One common approach to represent nonlinear movements is to encode the overall motion as a weighted combination of linear systems

$$\boldsymbol{\xi}^{\mathcal{O}} = \sum_{i}^{K} \overbrace{h_i(\boldsymbol{\xi}^{\mathcal{W}})}^{\substack{\text{scalar}\\ \text{weights}}} \overbrace{\Big[\boldsymbol{A}_i\,\boldsymbol{\xi}^{\mathcal{I}} + \boldsymbol{b}_i\Big]}^{\substack{\text{linear}\\ \text{subsystems}}}, \tag{2}$$

where the $\boldsymbol{\xi}^{\mathcal{I}} \in \mathbb{R}^{d^{\mathcal{I}}}$, $\boldsymbol{\xi}^{\mathcal{W}} \in \mathbb{R}^{d^{\mathcal{W}}}$ and $\boldsymbol{\xi}^{\mathcal{O}} \in \mathbb{R}^{d^{\mathcal{O}}}$ refer to input, weight and output variables. $\boldsymbol{A}_i \in \mathbb{R}^{d^{\mathcal{O}} \times d^{\mathcal{I}}}$ and $\boldsymbol{b}_i \in \mathbb{R}^{d^{\mathcal{O}}}$ define full matrices and offset vectors.

Examples of models that can be reformulated in this way are the *Gaussian mixture regression* (GMR) [16, 17], the *stable estimator of dynamical systems* (SEDS) [18], the *dynamic movement primitives* (DMP) [19, 20, 21], the *Takagi-Sugeno model* (TSM) [22], or computational models combining irrotational and solenoidal vector fields [23].

These methods can be classified and distinguished in the way $\boldsymbol{A}_i$ and $\boldsymbol{b}_i$ are estimated and constrained, in the representation of $\boldsymbol{x}$, or in the superposition mechanism used to combine the different subsystems through scalar weights h_i. Due to the generality of the representation, a wide range of possible extensions can be derived from this representation.

Fig. 1 presents an example of such encoding, where the activation functions $h_i(\boldsymbol{\xi}^{\mathcal{W}})$ are defined as *Gaussian mixture model* (GMM) weights

$$h_i(\boldsymbol{\xi}^{\mathcal{W}}) = \frac{\pi_i \; \mathcal{N}(\boldsymbol{\xi}^{\mathcal{W}} | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_k^K \pi_k \; \mathcal{N}(\boldsymbol{\xi}^{\mathcal{W}} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}, \text{ with } \mathcal{N}(\boldsymbol{\xi}^{\mathcal{W}} | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{\sqrt{(2\pi)^{d^{\mathcal{W}}} |\boldsymbol{\Sigma}_i|}} \exp\!\left(-\frac{1}{2} [\boldsymbol{\xi}^{\mathcal{W}} - \boldsymbol{\mu}_i]^{\top} \boldsymbol{\Sigma}_i^{-1} [\boldsymbol{\xi}^{\mathcal{W}} - \boldsymbol{\mu}_i] \right),$$

with priors π_i, centers $\boldsymbol{\mu}_i$ and covariances $\boldsymbol{\Sigma}_i$.

In Fig. 1, the movement to draw the letters 'ds' is encoded in a GMM of 4 components with an associated local flow field described by matrices $\boldsymbol{A}_i$ and vectors $\boldsymbol{b}_i$, see Eq. (2). We see that in the most generic case (matrices $\boldsymbol{A}_i$ with non-zero off-diagonal elements and non-zero vectors $\boldsymbol{b}_i$), each linear system can define various types of dynamics (curls, sinks, etc.). In this figure, two examples of superposition mechanisms are illustrated, relying on a GMM in time or space, in which the likelihood defines the space or time regions where each subsystem is active. In this example, we thus have either $\boldsymbol{\xi}^{\mathcal{I}} = \boldsymbol{\xi}^{\mathcal{W}} = \boldsymbol{x}$ and $\boldsymbol{\xi}^{\mathcal{O}} = \dot{\boldsymbol{x}}$ (top-right block), or $\boldsymbol{\xi}^{\mathcal{I}} = \boldsymbol{\xi}^{\mathcal{W}} = t$ and $\boldsymbol{\xi}^{\mathcal{O}} = \dot{\boldsymbol{x}}$ (bottom-right block), both retrieving similar movements for the same initialization and without perturbation.

The core idea of *dynamic movement primitives* (DMP) [19, 20] is to modulate a point-to-point spring-damper system with a nonlinear force represented as a weighted sum of learned constant force components f_i, namely

$$\tau \ddot{x} = \kappa^{\mathcal{P}} [x_T - x] - \kappa^{\mathcal{V}} \dot{x} + f(t), \quad \text{with } f(t) = \sum_{i=1}^{K} h_i(t) f_i, \tag{3}$$

where τ is a time constant. Instead of defining the activation functions $h_i(t)$ with time as a driving mechanism, a second dynamical system can alternatively be employed (for example, of the form $\tau \dot{s} = -\alpha s$) to provide additional flexibility with a parameterizable decay term. The resulting system is often formulated as

$$\tau \ddot{x} = \kappa^{\mathcal{P}} [x_T - x] - \kappa^{\mathcal{V}} \dot{x} + f(s), \quad \text{with } f(s) = s[x_T - x_0] \sum_{i=1}^{K} h_i(s) f_i,$$

to ensure spatial scaling properties, and to guarantee that the force vanishes at the end of the movement.

It was shown in [21] that DMP could be explained with a mechanical analogy by defining the force components f_i as additional virtual spring-damper systems $f_i = \kappa^{\mathcal{P}} (\mu_i^x - x) - \kappa^{\mathcal{V}} \dot{x}$. It adds local corrective terms that can swiftly react to perturbations introduced at reproduction time. It also moves the learning problem to the estimation of virtual attractor points μ_i^x. By adopting this formulation, the core mechanism of DMP can also be described with (2), see [24]. By considering that the final spring-damper in (3) is also estimated from the data, the matrices $\boldsymbol{A}_i$ and offset vectors $\boldsymbol{b}_i$ are constrained to have the form

$$\overbrace{\begin{bmatrix} \dot{\boldsymbol{x}} \\ \ddot{\boldsymbol{x}} \end{bmatrix}}^{\dot{\chi}} = \sum_i h_i \left(\overbrace{\begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \begin{bmatrix} -\kappa^{\mathcal{P}} & 0 \\ 0 & \ddots \end{bmatrix} & \begin{bmatrix} -\kappa^{\mathcal{V}} & 0 \\ 0 & \ddots \end{bmatrix} \end{bmatrix}}^{\boldsymbol{A}_i} \overbrace{\begin{bmatrix} \boldsymbol{x} \\ \dot{\boldsymbol{x}} \end{bmatrix}}^{\chi} + \overbrace{\begin{bmatrix} \mathbf{0} \\ \kappa^{\mathcal{P}} \boldsymbol{\mu}_i^{\boldsymbol{x}} \end{bmatrix}}^{\boldsymbol{b}_i} \right), \tag{4}$$

where the state $\chi = \left[\begin{smallmatrix} \boldsymbol{x} \\ \dot{\boldsymbol{x}} \end{smallmatrix} \right]$ has components $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ corresponding to position and velocity. $\mathbf{0}$ and $\boldsymbol{I}$ are null and identity matrices of corresponding sizes.

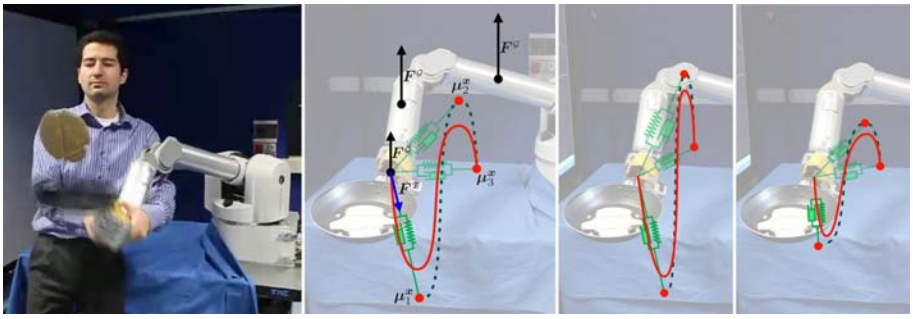


Figure 2: Schematic view of the learning process in the pancake flipping task. A weighted least-squares method is first used to fit an initial policy to the demonstration. The skill is then progressively refined through exploration in the policy parameters space.

DMPs are usually constrained to have fixed stiffness and damping scalar parameters (in (4), each variable acts as a separated system synchronized by shared weights h_i). We propose here to replace the diagonal matrix of constant scalar gains $\kappa^{\mathcal{P}}$ with learned full stiffness matrices $\boldsymbol{K}_i^{\mathcal{P}}$. This allows us to encapsulate the local synergies among the different variables in the policy, by constraining (2) to be in the form

$$\ddot{\boldsymbol{x}} = \sum_{i=1}^{K} h_i \Big[\boldsymbol{K}_i^{\mathcal{P}} [\boldsymbol{\mu}_i^{\boldsymbol{x}} - \boldsymbol{x}] - \kappa^{\mathcal{V}} \dot{\boldsymbol{x}} \Big]. \tag{5}$$

In the above equation, $\boldsymbol{\mu}_i$, $\boldsymbol{K}_i^{\mathcal{P}}$ and $\kappa^{\mathcal{V}}$ are respectively the attractor point, the stiffness and damping parameters of the i-th virtual spring-damper system. The consideration of full stiffness matrices $\boldsymbol{K}_i^{\mathcal{P}}$ instead of a constant scalar gain permits the adaptation of compliance behaviors by considering task-dependent constraints, see e.g. [25]. This choice is also driven by the importance of representing local coordination in motor control, see e.g. [26, 27, 28, 29, 30]. For example, in Fig. 1, the use of scalar gains would limit each local movement behavior to be of a single attractor type (uncorrelated flow field converging locally in straight line to the virtual attractor).

The formulation is compatible with attractor point control schemes suggesting that the central nervous system utilizes spring-like properties of the neuromuscular system in coordinating multi-degrees of freedom human limb movements [31], thus simplifying the control scheme to move along a trajectory by following intermediate equilibrium postures.

The mechanical analogy provides a compact and interpretable parameterization of the policy, and facilitates the transition from stiff trajectory skill acquisition to compliant behaviors learning. Compactness is important for exploration purpose, because it directly influences the number of trials required to learn a skill. Interpretability is important for the user to assess what the robot has learned and to scaffold the policy exploration process. Compactness and interpretability are also important factors for digital storage, analysis and skills manipulation/intervention purposes [32].

3.1. Pancake flipping experiment

Most robot skill acquisition approaches focussed on trajectory learning, but recently, a growing interest has emerged toward extending this learning problem to the joint acquisition of movement, coordination and compliance [3, 33, 34, 35, 6, 36].

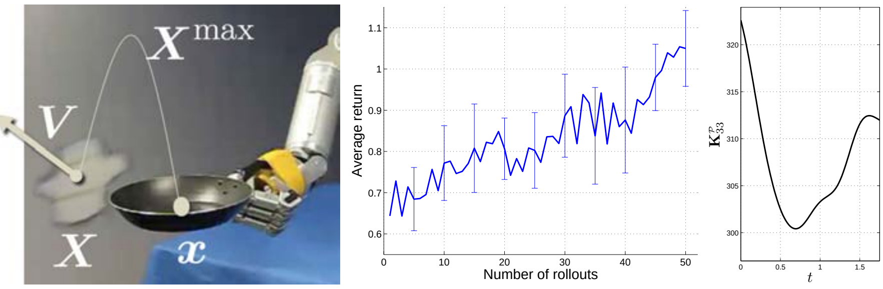


Figure 3: *Left:* Variables being used for the computation of the reward. *Center:* Evolution of the rewards over the trials. *Right:* Learned stiffness profile in the vertical direction.

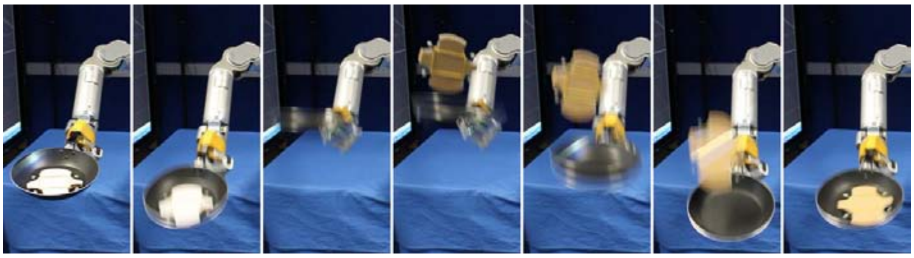


Figure 4: Reproduction of the skill after refinement of the movement with EM-based reinforcement learning.

Some skills remain difficult to transfer from demonstrations. This includes skills requiring highly dynamic movements, skills where the physical guidance of the robot can interfere with the tasks, or skills where the regularities in the demonstrations are not meaningful to extract task constraints efficiently.

We extend in this experiment the ball-in-a-cup task presented by Kober and Peters in [13] to a pancake flipping task, see also [37]. Learning such a skill dwells on the category of tasks that are difficult to transfer kinesthetically, not only because of the highly dynamic movement, but also because the demonstrations in the training set can be inconsistent. Since the task is difficult to execute, extracting regular patterns in the user's demonstrations becomes difficult.

This experiment shows that the compact encoding in (5) can be jointly used in an imitation and *reinforcement learning* (RL) context to acquire a skill requiring different levels of compliance along the movement. The main novelty with respect to [13] is that both the trajectory and the varying stiffness parameters are simultaneously refined by the robot in task space.

The experiment is implemented on a *Barrett WAM* torque-controlled 7 DOFs manipulator with a frying pan attached at the end-effector. The robot is gravity-compensated during demonstrations and reproductions. It is controlled by inverse dynamics solved with a recursive Newton-Euler formulation [38]. A gravity compensation force is added to each link, and a wrench command in Cartesian space is used to control the orientation of the end-effector through a resolved-acceleration control scheme. The joint forces $\boldsymbol{f}_i$ at each joint $i \in \{1, \ldots, 7\}$ are recursively calculated as

$$\boldsymbol{f}_i = \boldsymbol{f}_i^{\mathcal{A}} - \boldsymbol{f}_i^{\mathcal{E}} + \sum_{j \in c_i} \boldsymbol{f}_j,$$

where $\boldsymbol{f}_i^{A}$ is the net force acting on link i, $\boldsymbol{f}_j$ where $j \in c(i)$ are the forces transmitted by the previous links c_i in the kinematic chain, and $\boldsymbol{f}_i^{\mathcal{E}}$ are the external forces that take into consideration $\boldsymbol{f}^{\mathcal{T}}=[\boldsymbol{F}^{\mathcal{T}}, \boldsymbol{M}^{\mathcal{T}}]^{\top} \in \mathbb{R}^6$ (force and momentum required to accomplish the task) and $\boldsymbol{f}^{\mathcal{G}} = [\boldsymbol{F}^{\mathcal{G}}, \boldsymbol{0}]^{\top} \in \mathbb{R}^6$ (gravity compensation forces), see Fig. 2.

The skill is learned with policy parameters $\boldsymbol{\Theta}^{(n)}$ representing $\{\boldsymbol{K}_i^{\mathcal{P}}, \boldsymbol{\mu}_i^{x}\}$ in Eq. (5) at the n-th trial.[1] A single initial demonstration is provided by the user holding the frying pan and moving the robot to kinesthetically demonstrate the task. The policy parameters are initialized from this demonstration with a locally weighted learning procedure detailed in [36]. This initial demonstration is not sufficient to reproduce the task correctly.

Fig. 2 presents a schematic view of the policy learning problem. In this illustration, only three virtual attractors $\boldsymbol{\mu}_i^{x}$ are depicted (red dots), connected by virtual spring-damper systems to the end-effector. At the beginning of the movement, the first virtual spring shows the highest activation weight (the other springs are represented with semi-transparent colors). At each time step, the attractor resulting from the weighted combination of the three virtual attractors follows a continuous path represented by a green dotted line in the figure. The resulting path of the end-effector is represented with a thick red line. It is relevant to notice that the end-effector does not necessary reach the intermediate virtual attractors, due to the overlapping activation weights.

The goal of the task is to toss a pancake in the air and catch it after a half flip. An artificial pancake is used with 4 reflective passive markers to track its position and orientation. A model of 8 components is employed, empirically determined by examining the quality of the initial reproduction with different numbers of components. The reward function $r(\boldsymbol{\Theta})$ is defined as a weighted sum of criteria encouraging successful flipping and catching of the pancake

$$r(\boldsymbol{\Theta}) = w_1 \frac{\arccos(\boldsymbol{V}_0 \boldsymbol{V}^{\top})}{\pi} + w_2 \exp(-||\boldsymbol{X} - \boldsymbol{x}||) + w_3 \boldsymbol{X}_3^{\max},$$

computed at the moment when the pancake falls back down at a height Δ_h above the frying pan. $\boldsymbol{V}_0$ and $\boldsymbol{V}$ are the initial and final orientation vectors of the pancake (unit vector perpendicular to the pancake) at this height. $\boldsymbol{X}$ and $\boldsymbol{x}$ are respectively the pancake and frying pan position at this height. $\boldsymbol{X}_3^{\max}$ is the maximum altitude reached by the pancake during the toss, see Fig. 3-*left* for the variables involved in the computation of the rewards. The first term is maximized when the pancake falls back with a half-flip. The second term is maximized when the pancake lands in the center of the frying pan. The third term has been set to encourage higher throws. The weights have been set empirically in the experiment as $w_1 = w_2 = w_3 = 0.5$. The mass of the frying pan was not explicitly provided to the robot in this experiment, which means that the robot also needed to learn how to adapt the movement according to this weight.

Less than 100 rollouts were necessary to reproduce the task skillfully and consistently. Fig. 4 shows a reproduction result after 50 trials, with a policy initiated from a single demonstration.[2] The robot learned that at the beginning of the movement, it requires a certain level of stiffness to throw the pancake in the air, but that it can be more compliant after the throw to smoothly catch the pancake without making it bounce off the pan. Fig. 3 shows the evolution of the rewards and the resulting stiffness profile for the vertical direction.

3.2. Related work

This experiment follows other works tackling the problem of learning appropriate stiffness matrices to handle dynamic movements and/or contact with the environment. Rosenstein *et al*

[1] The exploration constrains $\boldsymbol{K}_i^{\mathcal{P}}$ to be symmetric semi-positive definite, and the difference of magnitudes in $\boldsymbol{K}_i^{\mathcal{P}}$ and $\boldsymbol{\mu}_i^{x}$ are taken into account through rescaling.

[2] In the first reproduction trial, the reproduced movement is not dynamic enough to let the pancake be tossed.

presented in [33] a robot weightlifting experiment in which an appropriate coordination of the joints exploiting the robot's intrinsic dynamics is searched through RL. Their work highlights the advantage of considering off-diagonal elements in gain matrices to enable active coupling of the individual joints. They showed through experiments that skillful movements that exploit dynamics are best acquired by first learning (or specifying) simple kinematic movement, and then letting the robot practice to transform that movement into dynamic solution with tighter coupling from the control system.

Kim *et al* proposed in [35] to use a recursive least-squares episodic natural actor-critic algorithm to discover optimal full stiffness matrices in task space with simulated experiments such as opening a door, catching a ball, and reaching a point in an unknown force field.

Pardo *et al* presented in [34] a simulated experiment on quadruped and bipedal robots to learn motor coordination and joint synergies in rest-to-rest movements. They started from a basic decoupled representation of the movement by considering point-to-point movements driven by a proportional-derivative controller. RL was then used to learn how to coordinate the set of variables which were originally decoupled.

The approaches above differ to ours in the sense that the movement is predefined and the impedance/coordination is learned with respect to this reference trajectory or point-to-point movement, while we tackle the problem of learning the movement and stiffness/coordination concurrently.

The closest approach to ours was presented by Buchli *et al* in [6]. The authors proposed to use PI^2 (together with a DMP representation of the trajectory) to learn movements while optimizing impedance parameters. The approach was demonstrated in a light switch toggling experiment and in simulated experiments of door pushing. Their approach optimizes the trajectory (in joint space or task space) and time-varying impedance parameters in joint space by considering each variable separately. Namely, a diagonal stiffness matrix in joint space is searched, resulting in a full stiffness matrix in task space after projection through the Jacobian.

We propose instead to directly estimate the full stiffness matrix in task space, and use it to compute a force command (being converted to torque commands through the Jacobian). This approach allows the user to exploit the robot's redundant degrees of freedom while it reproduces the skill with a desired stiffness and movement behaviors in task space. For example, if the user and the robot share the same workspace, it becomes possible for the user to put the robot in a different kinematic configuration to facilitate his/her task in close proximity to the robot. In other words, the nullspace is explicitly specified by assuming that the relevant parameters for the compliance behavior of the robot are in task space. This design choice aims at exploiting the nullspace in future work to concurrently learn stiffness constraints in joint and task space (e.g., to search for a preferred robot posture while manipulating object in task space).

4. Exploitation of covariance information in RL

This section explores how EM learning strategies can be exploited to stochastically sample for new policies in an adaptive manner.

Exploration procedures can sometimes consider simple forms of noise, by introducing decaying (or constant) exploration terms, or noise generated independently for each variable (without covariation). On one hand, large exploration noise can lead to faster convergence due to greater changes of the mean policy. On the other hand, low exploration noise can converge to more accurate solutions, while avoiding bringing the robot into unsafe regimes. The exploration thus has to be sufficiently rich to converge in a reasonable amount of time and avoid getting stuck in local minima. As suggested by Peters and Schaal [12], the EM process can also be used to optimize exploration noise together with policy parameters, which is also exploited in stochastic optimization and evolutionary search [8, 9].

The exploration noise can be expressed in the form of a covariance matrix and updated with

$$\mathbf{\Sigma}^{(n)} = \frac{\sum_{m}^{M} r(\mathbf{\Theta}_m) \left[\mathbf{\Theta}_m - \mathbf{\Theta}^{(n-1)}\right]\left[\mathbf{\Theta}_m - \mathbf{\Theta}^{(n-1)}\right]^{\top}}{\sum_{m}^{M} r(\mathbf{\Theta}_m)} + \mathbf{\Sigma}_0, \tag{6}$$

where $\mathbf{\Sigma}_0$ is a regularization term (here, diagonal covariance matrix) corresponding to a minimum exploration noise used to avoid premature convergence.

Equations (1) and (6) form at each iteration a multivariate normal distribution with a center and a covariance matrix in the policy parameters space. This property can be exploited in different ways. First, such representation can be exploited for the search of solutions based on both high rewards and high tolerances to errors. In other words, by searching for tolerant solutions that are least sensitive to motor control errors and external perturbations. This representation also conveys important information about the neighborhood (e.g., shape, size, curvature, principal directions) in which the policy parameters can be modulated to search for new policies.

In the case of human movements, it has been shown that redundancy provides a way to cope with noise at a motor level by channeling it in directions that have minimal effect on achieving the task goal [39]. Skilled performers can take advantage of this redundancy and align their actions with the solution manifold corresponding to a given task goal, i.e., the space in which noise and variability have little or no effect on the end result.

It is well known that each repetition of an act involves unique, non-repetitive neural and motor patterns [40]. Instead of viewing variability as a nuisance when collecting data, it is viewed as a central tool to study the organization of the system producing voluntary movements. In everyday tasks such as writing or drinking from a glass, elements such as the griping force have relatively weak constraints (it should only be in a range that prevents the object from falling or being damaged), while other elements need to be controlled more precisely (to write legibly or to avoid spilling the content of the glass). The term *uncontrolled manifold* is used to refer to the core strategy in which some elements can be less controlled as long as they remain within an acceptable range [28]. It follows the principle of *minimal interaction*, stating that if an element produces an error in the common output, other elements can change their contributions to minimize this error without requiring complex corrective actions from the controller.

We take inspiration from the proposal addressed in [39] that variability may offer a way to quantify error tolerance via the shape of the result function. Indeed, one obvious reason to prefer some locations over others in the policy parameters space is the sensitivity of the result to variability (tolerance to errors). Covariance information in EM-based RL can be used to assess such error tolerance. The immediate neighborhood of a solution manifold may have different curvature for different local optima, making some high rewards regions in the policy parameters space more tolerant to errors than others. A solution manifold can for example be characterized by a continuous portion of space in which the reward is maximum, where the local spread of the region determines the tolerance to errors of this solution.

Covariance information can here be exploited in several ways. In the next section, we show a natural extension of the EM-based search procedure to mixtures of policy subspaces. Namely, to the search of a control policy solution landscape by fitting a density function that does not (necessary) have a single optimum. More complex solution space can in this way be approximated by multimodal distributions modeled in an adaptive and incremental manner through stochastic sampling.

4.1. Multi-optima policy search

The search of a single optimal control policy can remain inappropriate for a wide range of applications, see e.g. [41]. For example, by observing the best sprinters in the world, studies such as [42] reveal that elite athletes can practice and improve their running performance by relying on different policies, favoring two different categories of local optima corresponding to either long strides or high cadence.

An extension of EM-based RL to the autonomous search of multiple local optima in the policy parameters space can be derived by extending Eqs (1) and (6) to a *Gaussian mixture model* (GMM) in which a set of normal distributions characterized by centers $\boldsymbol{\mu}_i$, covariance matrices $\boldsymbol{\Sigma}_i$ and priors π_i are iteratively refined. Each $\boldsymbol{\mu}_i$ represents the best local guess of the policy parameters that leads to high rewards, while $\boldsymbol{\Sigma}_i$ represents the spread of the region. The prior (or mixing factor) π_i is associated with each Gaussian to encode the importance of each local solution subspace.

The use of GMM was also recently explored by Kobilarov in [43] for stochastic sampling in CEM. The author showed that, for exploration in a parameterized trajectory space, the use of GMM could improve the performance of sampling-based motion planning with a more global exploration of feasible solutions.

The GMM representation can be useful in two situations: 1) when several local optima are separated in the policy parameters space (see Fig. 5); and 2) when the local solution space has a complex or asymmetric shape in the policy parameters space that cannot be approximated efficiently by a single Gaussian.

In the first case (Fig. 5), the search for multi-optima policy subspaces can be exploited to evaluate different solutions and select the most appropriate with respect to the current situation (context-dependent selection process). Such a choice may for example depend on external factors such as space restriction, occlusion, injured articulation or fatigued muscles (or equivalently, broken or overheated motors). It also allows the robot to robustly adapt to progressively changing environment (drifting reward functions). In this case, the system can keep track of regions that might currently have slightly lower reward but that remain of interest, because they can potentially lead to optimal solutions in the future, or because they can be used as an alternative option if the ”best” policy is unavailable or is too risky to be used in the current context.

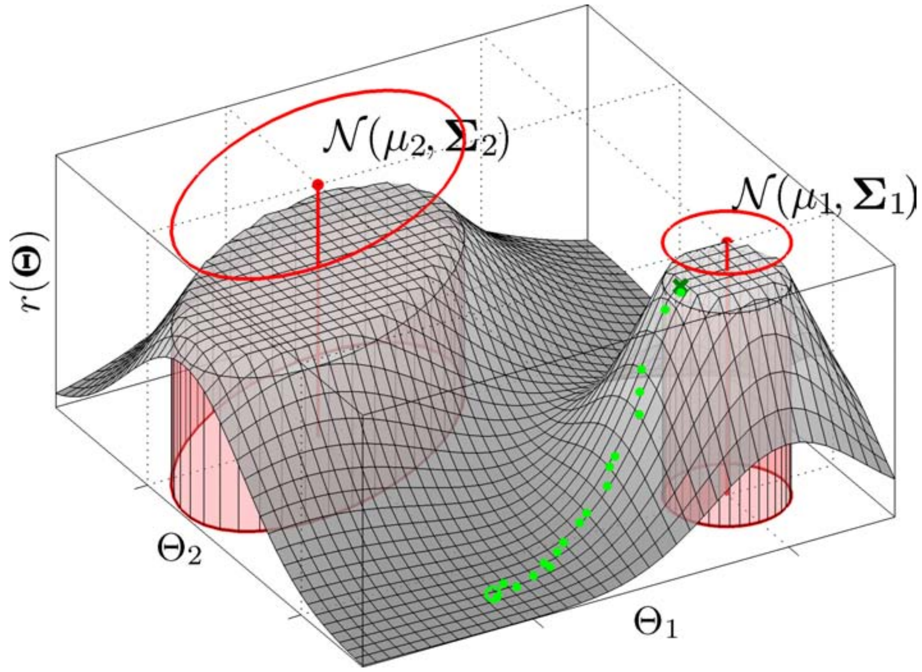
To extend the update process in Eqs (1) and (6) to the mixture case, the mixing weights do not only consider the reward, but also the probability to belong to one of the mixture components. Namely, for each $i \in \{1, \ldots, K\}$, we define the EM procedure

E-step:

$$h_i(\boldsymbol{\Theta}_m) = \frac{\pi_i \; \mathcal{N}\Big(\boldsymbol{\Theta}_m | \, \boldsymbol{\mu}_i^{(n-1)}, \boldsymbol{\Sigma}_i^{(n-1)}\Big)}{\sum_k^K \pi_k \; \mathcal{N}\Big(\boldsymbol{\Theta}_m | \, \boldsymbol{\mu}_k^{(n-1)}, \boldsymbol{\Sigma}_k^{(n-1)}\Big)}.$$

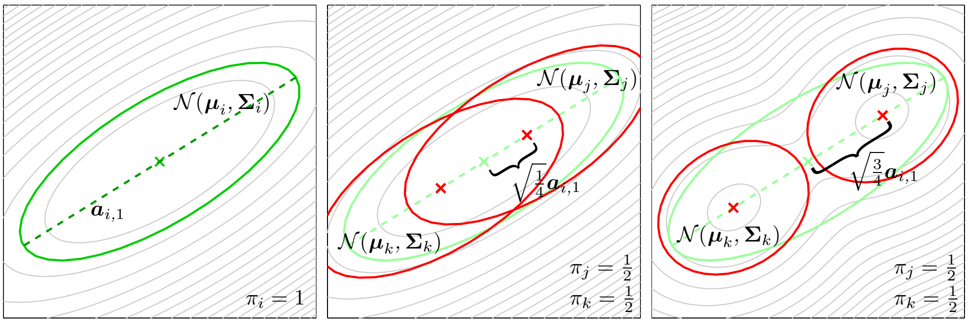
M-step:

$$\boldsymbol{\mu}_i^{(n)} = \boldsymbol{\mu}_i^{(n-1)} + \frac{\sum\limits_m^M r(\boldsymbol{\Theta}_m) h_i(\boldsymbol{\Theta}_m) \left[\boldsymbol{\Theta}_m - \boldsymbol{\mu}_i^{(n-1)}\right]}{\sum\limits_m^M r(\boldsymbol{\Theta}_m) h_i(\boldsymbol{\Theta}_m)},$$



$\mathcal{N}(\mu_2, \mathbf{\Sigma}_2)$
$\mathcal{N}(\mu_1, \mathbf{\Sigma}_1)$
$r(\mathbf{\Theta})$
Θ_2
Θ_1

Figure 5: Illustration of an exploration problem characterized by several local optima. The task, described by two policy parameters $\boldsymbol{\Theta} = [\Theta_1, \Theta_2]^\top$, has a solution landscape represented by a shaded surface with elevation and color intensity proportional to the rewards $r(\boldsymbol{\Theta})$. This policy-reward mapping is initially unknown to the robot. In this illustrative example, two local optima could be found by the robot, characterized by centers $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2\}$ and covariance matrices $\{\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2\}$. A standard gradient-ascent procedure starting from the green circle can converge to one of the two peaks and possibly stops after attaining a local maximum (green cross). We are interested in exploration procedures that are approximating regions of high rewards with a density function, such that the algorithm does not only detect a peak and stops, but continues instead the exploration to provide more information about the viability and spread of the solution subspace. In this example, the reward $r(\boldsymbol{\mu}_1)$ is slightly higher than $r(\boldsymbol{\mu}_2)$. However, $\boldsymbol{\mu}_2$ can be considered as a safer choice for the policy because the larger covariance $\boldsymbol{\Sigma}_2$ results in higher tolerance to errors. The shape of the Gaussians can similarly provide information about the correlations between the policy parameters, which can be used to select appropriate context-dependent actions.



$\mathcal{N}(\mu_i, \Sigma_i)$
$a_{i,1}$
$\pi_i = 1$
$\mathcal{N}(\mu_j, \Sigma_j)$
$\sqrt{\frac{1}{4}} a_{i,1}$
$\mathcal{N}(\mu_k, \Sigma_k)$
$\pi_j = \frac{1}{2}$
$\pi_k = \frac{1}{2}$
$\mathcal{N}(\mu_j, \Sigma_j)$
$\sqrt{\frac{3}{4}} a_{i,1}$
$\mathcal{N}(\mu_k, \Sigma_k)$
$\pi_j = \frac{1}{2}$
$\pi_k = \frac{1}{2}$

Figure 6: Gaussian splitting process in equal proportion. *Left:* Original Gaussian with principal axis $\boldsymbol{a}_{i,1}$ depicted in a dashed line and equiprobability lines in the background depicted in light grey color. *Center and right:* Influence of the parameter u on the splitting process ($u = 0$ would correspond to a perfect match of equiprobability lines).

$$\boldsymbol{\Sigma}_i^{(n)} = \frac{\sum_m^M r(\boldsymbol{\Theta}_m) h_i(\boldsymbol{\Theta}_m) \Big[\boldsymbol{\Theta}_m - \boldsymbol{\mu}_i^{(n-1)}\Big]\Big[\boldsymbol{\Theta}_m - \boldsymbol{\mu}_i^{(n-1)}\Big]^{\top}}{\sum_m^M r(\boldsymbol{\Theta}_m) h_i(\boldsymbol{\Theta}_m)} + \boldsymbol{\Sigma}_0,$$

$$\pi_i^{(n)} = \frac{\sum_m^M r(\boldsymbol{\Theta}_m) h_i(\boldsymbol{\Theta}_m)}{\sum_k^K \sum_m^M r(\boldsymbol{\Theta}_m) h_k(\boldsymbol{\Theta}_m)}. \tag{7}$$

In the above equations, $\{\boldsymbol{\Theta}_m\}_{m=1}^M$ is the ordered set of the best policy parameters for each component i, with $r(\boldsymbol{\Theta}_1)h_i(\boldsymbol{\Theta}_1) \geq r(\boldsymbol{\Theta}_2)h_i(\boldsymbol{\Theta}_2) \geq \ldots \geq r(\boldsymbol{\Theta}_M)h_i(\boldsymbol{\Theta}_M)$. M is the parameter of the importance sampler (M can be set to the current number of trials to disable the elite strategy). $\boldsymbol{\Sigma}_0$ is the regularization term to avoid early convergence.

The above GMM formulation requires some form of online model selection. We employ here a simple heuristic by initially considering the policy parameters landscape as unimodal (a single Gaussian distribution is used, as in PoWER, CEM and CMA-ES), and by progressively adapting the probabilistic distribution based on the exploration results. The GMM split&merge algorithm proposed by Zhang *et al* [44] is used together with the update rule in Eq. (7) to autonomously estimate a possibly multi-peaked solution space. The algorithm is described below.

4.1.1. GMM split&merge algorithm

For the search of a policy $\boldsymbol{\Theta}$, the gradual resolution decrease or increase of the solution manifold is achieved by merging and splitting the Gaussian distributions and associated priors in the current GMM. Each Gaussian can be split if the addition of one component has a significant impact on the likelihood. Similarly, two Gaussians are merged if the removal of one component only slightly decreases the likelihood.

The merging of two Gaussians j and k into a Gaussian i is characterized by $\pi_i = \pi_j + \pi_k$ and $\pi_i \mathcal{P}(\boldsymbol{\Theta}|i) = \pi_j \mathcal{P}(\boldsymbol{\Theta}|j) + \pi_k \mathcal{P}(\boldsymbol{\Theta}|k)$. It can be shown that the closest result satisfies the relations (see [44] for details)

$$\pi_i \boldsymbol{\mu}_i = \pi_j \boldsymbol{\mu}_j + \pi_k \boldsymbol{\mu}_k, \quad \text{and} \quad \pi_i(\boldsymbol{\Sigma}_i + \boldsymbol{\mu}_i \boldsymbol{\mu}_i^{\top}) = \pi_j(\boldsymbol{\Sigma}_j + \boldsymbol{\mu}_j \boldsymbol{\mu}_j^{\top}) + \pi_k(\boldsymbol{\Sigma}_k + \boldsymbol{\mu}_k \boldsymbol{\mu}_k^{\top}). \tag{8}$$

Algorithm 1 Multi-optima policy search with GMM split&merge algorithm.

```
for $n \leftarrow 1$ to $N$ do                                   ▹ Loop for each trial $n$
    $\boldsymbol{\Theta}_n \leftarrow$ randomSampling($\hat{\Omega}$)              ▹ Random sampling of a $n$-th point
    $\hat{\Omega} \leftarrow$ EM($\hat{\Omega}$,$\boldsymbol{\Theta}$)    ▹ Refine GMM (policy parameters and exploration noise)
    for $i \leftarrow 1$ to $K$ do                  ▹ Compute splitting candidates for all Gaussians
        $\Omega_i \leftarrow$ splitGaussian($\hat{\Omega}$,i)
        $L^{\mathcal{S}}_i \leftarrow$ evaluateLikelihood($\Omega_i$)
    end for
    $\hat{L} \leftarrow$ evaluateLikelihood($\hat{\Omega}$)
    if $(\max(\boldsymbol{L}^{\mathcal{S}}) - \hat{L}) > T^{\mathcal{S}}$ then
        $\hat{\Omega} \leftarrow \Omega_{\arg\max(\boldsymbol{L}^{\mathcal{S}})}$                  ▹ Split Gaussian (w.r.t. threshold)
    end if
    for $i \leftarrow 1$ to $K-1$ do     ▹ Compute merging candidates for all pairs of Gaussians
        for $j \leftarrow i+1$ to $K$ do
            $\Omega_{ij} \leftarrow$ mergeGaussians($\hat{\Omega}$,i,j)
            $L^{\mathcal{M}}_{ij} \leftarrow$ evaluateLikelihood($\Omega_{ij}$)
        end for
    end for
    $\hat{L} \leftarrow$ evaluateLikelihood($\hat{\Omega}$)
    if $(\hat{L} - \max(\boldsymbol{L}^{\mathcal{M}})) < T^{\mathcal{M}}$ then
        $\hat{\Omega} \leftarrow \Omega_{\arg\max(\boldsymbol{L}^{\mathcal{M}})}$                  ▹ Merge Gaussians (w.r.t. threshold)
    end if
end for
```

Solving the split equations is an ill-posed problem (fewer equations than unknowns), which can however be approximated through *singular value decomposition* (SVD) of the covariance matrices. By defining the set of parameters $l \in \{1, 2, \ldots, N\}$, $\alpha \in [0, 1]$, $\beta \in [0, 1]$ and $u \in [0, 1]$, a Gaussian $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ with prior π_i is split into two Gaussians $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ and $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ with parameters

$$\begin{aligned}
\pi_j &= \pi_i \alpha, & \pi_k &= \pi_i (1-\alpha), \\
\boldsymbol{\mu}_j &= \boldsymbol{\mu}_i - \sqrt{\frac{\pi_k}{\pi_j}}\, u\, \boldsymbol{a}_{i,l}, & \boldsymbol{\mu}_k &= \boldsymbol{\mu}_i + \sqrt{\frac{\pi_j}{\pi_k}}\, u\, \boldsymbol{a}_{i,l}, \\
\boldsymbol{\Sigma}_j &= \frac{\pi_k}{\pi_j}\boldsymbol{\Sigma}_i + (\beta - \beta u^2 - 1)\frac{\pi_i}{\pi_j}\boldsymbol{a}_{i,l}\boldsymbol{a}_{i,l}^{\top} + \boldsymbol{a}_{i,l}\boldsymbol{a}_{i,l}^{\top}, & \boldsymbol{\Sigma}_k &= \frac{\pi_j}{\pi_k}\boldsymbol{\Sigma}_i + (\beta u^2 - \beta - u^2)\frac{\pi_i}{\pi_k}\boldsymbol{a}_{i,l}\boldsymbol{a}_{i,l}^{\top} + \boldsymbol{a}_{i,l}\boldsymbol{a}_{i,l}^{\top}.
\end{aligned} \tag{9}$$

In the above equations, $\boldsymbol{\Sigma}_i = \boldsymbol{A}_i \boldsymbol{A}_i^{\top}$ represents the ordered eigencomponents decomposition of $\boldsymbol{\Sigma}_i$ with $\boldsymbol{A}_i = [\boldsymbol{a}_{i,1}, \boldsymbol{a}_{i,2}, \ldots, \boldsymbol{a}_{i,D}]$. The parameters $\alpha = \beta = \frac{1}{2}$, $u = \sqrt{\frac{3}{4}}$, $l = 1$ and $D = 1$ are used in our application, corresponding to equally weighted splitting operations by following the principal direction of the Gaussians. Fig. 6 presents an illustration of the split algorithm.

Algorithm 1 presents the pseudocode of the learning process. $\hat{\Omega} = \{\pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K}$ represents the current GMM (initialized with a single Gaussian). Ω_i and Ω_{ij} represent candidate GMMs for the splitting and merging operations. The *randomSampling()* function generates a datapoint $\boldsymbol{\Theta}_n$ from the distribution $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, with i stochastically drawn from the distribution $\boldsymbol{\pi}$. The *EM()* function is the weighted expectation-maximization algorithm presented in Eq. (7), used to refine the current GMM $\hat{\Omega}$ to fit the training dataset $\boldsymbol{\Theta}$. The *splitGaussian()* and *mergeGaussians()*

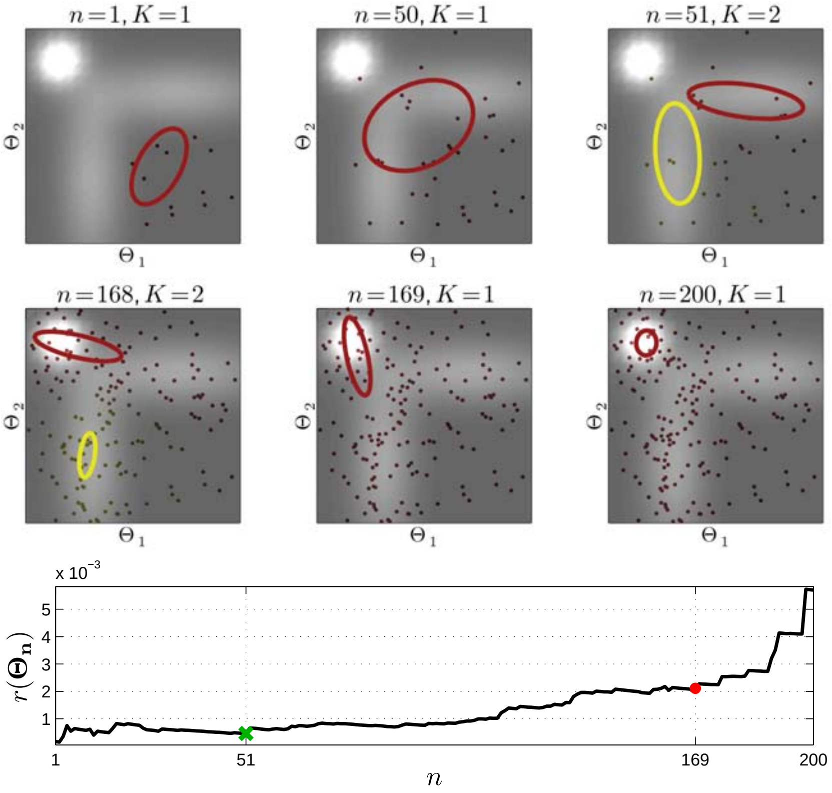


Figure 7: Multi-optima policy search with the EM-based GMM learning mechanism with an on-line incremental determination of the number of peaks. The green cross and red point in the bottom graph respectively represent split and merge operations occurring during the exploration (with corresponding trials in the first two rows).

functions are defined in Eqs (8) and (9). The *evaluateLikelihood()* function computes the average log-likelihood of N weighted datapoints with $\frac{1}{\sum_n^N r(\boldsymbol{\Theta}_n)} \sum_{n=1}^N r(\boldsymbol{\Theta}_n) \log\left(\sum_{k=1}^K \pi_k \mathcal{N}(\boldsymbol{\Theta}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)$. $T^{\mathcal{S}}$ and $T^{\mathcal{M}}$ are split and merge likelihood thresholds empirically set in our experiments to $2 \cdot 10^{-4}$ and $2 \cdot 10^{-6}$.

4.2. Multi-optima policy search experiments

In order to study the multi-optima policy search problem, we employ in this section a simplified policy representation to facilitate the analysis and visualization of the learning behavior. Similarly to a bandit-problem in RL, we will consider here policies described as a single state and continuous actions composed of two parameters that can be visualized in two-dimensional graphs.

Two experiments are presented where the *split&merge* algorithm is incorporated in the policy search procedure. In the first experiment, a solution landscape in the policy parameters space is simulated as the weighted sum of Gaussians $r(\boldsymbol{\Theta}) = \sum_{i=1}^{3} \pi_i \mathcal{N}(\boldsymbol{\Theta}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, with parameters $\pi_1 = 0.4$, $\boldsymbol{\mu}_1 = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$, $\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix}$, $\pi_2 = 0.4$, $\boldsymbol{\mu}_2 = \begin{bmatrix} 7 \\ 7 \end{bmatrix}$, $\boldsymbol{\Sigma}_2 = \begin{bmatrix} 9 & 0 \\ 0 & 1 \end{bmatrix}$, $\pi_3 = 0.2$, $\boldsymbol{\mu}_3 = \begin{bmatrix} 1.5 \\ 8.5 \end{bmatrix}$ and $\boldsymbol{\Sigma}_3 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$, see

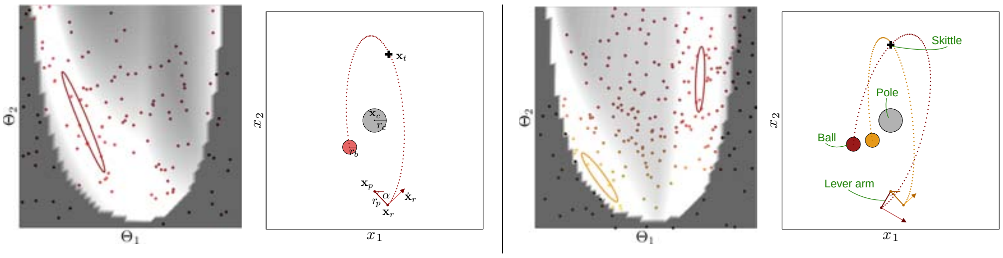


Figure 8: In a simulated table skittle game, the number of options in the policy parameters space can change depending on the position of the skittle. After 200 iterations, the algorithm finds either a unimodal (left) or bimodal distribution (right) to approximate the policy.

Fig. 7. The reward values are represented by shades of gray in the policy parameters space (the lighter the color, the higher the reward). The landscape corresponds to two spatially separated regions with equal medium-level rewards, and a region with significantly higher reward. It has the form of two horizontally and vertically oriented elongated hills ($r=0.02$, in gray shades) and a dominant peak on the top-left corner ($r=0.08$, in white shades). The search for this unknown solution is initiated with a policy search in the bottom-right corner of the policy parameters space.

A representative experimental run of the algorithm is presented in Fig. 7. The first two rows show relevant iterations of the RL process, and the last row shows the evolution of the reward. From $n=1$ to $n=50$, the search is displaced towards the center of the policy parameters landscape where the rewards are higher than in the bottom-right corner. A split of the Gaussian occurs at $n = 51$, when the system detects that the distribution is likely not unimodal and could be represented by two Gaussians (the two hills in the middle of the image). In the next hundred iterations, one of the two policy subspace (represented by the red Gaussian) discovers the top-left peak which has a higher reward value than the two hills. At $n = 169$, the policy subspace with the weakest reward (yellow Gaussian) becomes superfluous in comparison with the other region with significantly higher reward, resulting in the merging of the two Gaussians. Finally, after 200 iterations, the system correctly converges to the single peak solution corresponding to the only relevant optimal policy in this problem.

The second experiment considers the simulation of a table-top skittle game similar to the one presented in [45]. In the center of the board stands a pole with a ball suspended by a string from its top, represented by a gray circle in Fig. 8. The ball (red circle) is swung around the pole in order to hit a skittle (black '+' sign). The elliptic trajectory of the ball is generated by a 2D model in which the ball is attached by a massless spring to the center post. The swinging motion is damped to approximate realistic behaviors. The trajectory of the ball is determined by $\boldsymbol{x}(t) = \boldsymbol{A}\sin(\omega t + \boldsymbol{\phi})\exp(-\frac{t}{\tau})$.

The policy parameters are defined by $\boldsymbol{\Theta} = \left[\begin{smallmatrix}\alpha\\ \dot{\alpha}\end{smallmatrix}\right]$. The position and velocity of the ball at release are defined as $\boldsymbol{x}_r = \boldsymbol{x}_p + r_p \left[\begin{smallmatrix}\cos(\alpha)\\ -\sin(\alpha)\end{smallmatrix}\right]$ and $\dot{\boldsymbol{x}}_r = \dot{\alpha} r_p \left[\begin{smallmatrix}\sin(\alpha)\\ \cos(\alpha)\end{smallmatrix}\right]$, see the lever arm depicted in Fig. 8. The energy consists of a kinetic and potential component formulated as $\boldsymbol{E} = \frac{1}{2}(m\dot{\boldsymbol{x}}_r + k\boldsymbol{x}_r)$. The amplitudes and the phases are defined as $\boldsymbol{A} = \sqrt{\frac{2}{k}\boldsymbol{E}}$ and $\boldsymbol{\phi} = \arcsin(\frac{\boldsymbol{x}_r}{\boldsymbol{A}})$. The closest point to the target is labeled $\boldsymbol{x}_e$, with associated distance $d = \|\boldsymbol{x}_e - \boldsymbol{x}_t\|$. The reward function is defined by $r(\boldsymbol{\Theta}) = \exp(-\lambda d)$, where λ is a constant factor. The static parameters used in the simulation are $m=0.1$ kg, $k=1$ N/m, $\tau=20$ sec, $\omega=\sqrt{\frac{k}{m}-\frac{1}{\tau^2}}=3.16$ rad/sec, $\boldsymbol{x}_c=\left[\begin{smallmatrix}0\\ 0\end{smallmatrix}\right]$ m, $\boldsymbol{x}_p=\left[\begin{smallmatrix}0\\ -1.5\end{smallmatrix}\right]$ m, $\boldsymbol{x}_t=\left[\begin{smallmatrix}0.3\\ 1.4\end{smallmatrix}\right]$ m, $r_c=0.25$ m, $r_p=0.4$ m, $r_b=0.05$ m.

An interesting characteristic of this setup is that the number of peaks, shape and size of the highly rewarded regions in policy parameters space change with respect to the position of the skittle, see Fig. 8.

The results of a typical run of the experiment are presented for two different positions of the skittle. The first and third graphs in Fig. 8 show the solution landscape associated with the position of the skittle (the lighter the color, the higher the reward), created by computing the rewards throughout the policy parameters space. The second and fourth graphs show the results of the policy search process. After 200 iterations, the system correctly extracted the places and spreads in the policy parameters space that are successful for the task. For some positions of the skittle, a single Gaussian could approximate this shape (*left*), while in some other configuration, two Gaussians were required to approximate the distribution, corresponding to two different policy options (*right*).

Discovering a policy that is tolerant to errors is important in such skills. Human studies show that early attempts at playing the table skittle game can converge to an initial solution that can, with more substantial training, be shifted towards a solution subspace that is more robust to perturbations [39]. Inspired by these studies, we take here the perspective that instead of a single point, the GMM representation can approximate the local shape of several optimal regions, whose covariance information is not only used to generate stochastic samples, but can also be exploited to identify subspaces in the policy parameters space characterized by high reward and high tolerances to errors (see also illustrative example in Fig. 5).

The results of this experiment are encouraging, showing that the *split&merge* method performs well for simple policies. Further work will however be required to evaluate the performance of the approach for more complex policies such as in the pancake flipping scenario. A potential weakness of the current method, that will require further studies, concerns the threshold parameters that need be set by the experimenter. Future work will study the influence of these parameters on the final learning results, and possible ways of automatically setting them (e.g., with Bayesian information criterion). We plan in future work to contrast the proposed incremental *split&merge* solution with complementary methods for the online estimation of the model structure. In particular, we will investigate the use of Dirichlet processes and spectral clustering methods to estimate the number of peaks in an online manner.

5. Future work and conclusion

We presented two applications of EM-based RL for the search of parameterized policies. The first showed that exploration in the policy parameters space can be exploited to refine rich motor skills encapsulating movement and compliance behaviors. The second emphasized the double role of covariance information for stochastically sampling new policies and for modeling the space of possible policies. An extension of EM-based RL to multi-optima policy search is proposed by exploiting this capability.

The process of acquiring and refining skills in humans requires a savant combination of imitation, exploration, evaluation of performance and practice. In the experiments presented in this article, we emphasized the importance of providing the robot with self-improvement capabilities (by exploring new variants or refining a set of existing solutions). But in other situations, it is beneficial to provide (or let the robot request) new demonstrations. Often, skill acquisition in robots is studied by concentrating on a single learning aspect. Instead, learning in humans is tributary to the interconnections between these mechanisms. The core EM strategy of direct policy search in policy parameters space introduces possible connections with imitation. It corresponds to copying (at a certain extent defined by the covariance) the best trials obtained so far, in a proportion defined by their respective rewards.

The recent advances in robotics require us to consider these interconnections of imitation and practice, and to provide bi-directional teaching interfaces where the user can be part of the loop in robot learning. Possible extensions include mechanisms standing at the border of social learning and reinforcement learning. An example is the use of external rewards to complement internal rewards, where policies can be assessed by self-evaluation and rewarding signals from a human teacher [46]. Similarly, the exploration noise in the RL process should not be restricted to a single source. In the context of imitation, scaffolding techniques are used to speed up learning by modifying the environment or putting the robot in situations of increasing complexity. Such strategies could be applied to a reinforcement learning context by studying how the exploration noise can be modulated through human assistance.

References

[1] A. Billard, S. Calinon, R. Dillmann, S. Schaal, Robot programming by demonstration, in: B. Siciliano, O. Khatib (Eds.), Handbook of Robotics, Springer, Secaucus, NJ, USA, 2008, pp. 1371–1394.

[2] B. D. Argall, S. Chernova, M. Veloso, B. Browning, A survey of robot learning from demonstration, Robot. Auton. Syst. 57 (5) (2009) 469–483.

[3] T. Hulin, V. Schmirgel, E. Yechiam, U. E. Zimmermann, C. Preusche, G. Poehler, Evaluating exemplary training accelerators for Programming-by-Demonstration, in: Proc. IEEE Intl Symp. on Robot and Human Interactive Communication (RO-MAN), 2010, pp. 440–445.

[4] J. Peters, S. Schaal, Natural actor-critic, Neurocomput. 71 (7-9) (2008) 1180–1190.

[5] E. Theodorou, J. Buchli, S. Schaal, A generalized path integral control approach to reinforcement learning, J. Mach. Learn. Res. 11 (2010) 3137–3181.

[6] J. Buchli, F. Stulp, E. Theodorou, S. Schaal, Learning variable impedance control, Intl Journal of Robotics Research 30 (7) (2011) 820–833.

[7] T. Rueckstiess, F. Sehnke, T. Schaul, D. Wierstra, Y. Sun, J. Schmidhuber, Exploring parameter space in reinforcement learning, Paladyn. Journal of Behavioral Robotics 1 (1) (2010) 14–24.

[8] D. P. Kroese, R. Y. Rubinstein, The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning, Springer, 2004.

[9] N. Hansen, The CMA evolution strategy: A comparing review, in: J. Lozano, P. Larranaga, I. Inza, E. Bengoetxea (Eds.), Towards a New Evolutionary Computation, Vol. 192 of Studies in Fuzziness and Soft Computing, Springer Berlin / Heidelberg, 2006, pp. 75–102.

[10] R. J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Mach. Learn. 8 (3-4) (1992) 229–256.

[11] N. Vlassis, M. Toussaint, G. Kontes, S. Piperidis, Learning model-free robot control by a Monte Carlo EM algorithm, Autonomous Robots 27 (2009) 123–130.

[12] J. Peters, S. Schaal, Using reward-weighted regression for reinforcement learning of task space control, in: Proc. IEEE Intl Symp. on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), 2007, pp. 262–267.

[13] J. Kober, J. Peters, Imitation and reinforcement learning: Practical algorithms for motor primitives in robotics, IEEE Robotics and Automation Magazine 17 (2) (2010) 55–62.

[14] P. Dayan, G. E. Hinton, Using expectation-maximization for reinforcement learning, Neural Comput. 9 (2) (1997) 271–278.

[15] E. Todorov, Efficient computation of optimal actions, Proc. of the National Academy of Sciences of the United States of America 106 (28) (2009) 11478–11483.

[16] S. Calinon, F. Guenter, A. Billard, On learning, representing and generalizing a task in a humanoid robot, IEEE Trans. on Systems, Man and Cybernetics, Part B 37 (2) (2007) 286–298.

[17] M. Hersch, F. Guenter, S. Calinon, A. Billard, Dynamical system modulation for robot learning via kinesthetic demonstrations, IEEE Trans. on Robotics 24 (6) (2008) 1463–1467.

[18] S. M. Khansari-Zadeh, A. Billard, Learning stable non-linear dynamical systems with Gaussian mixture models, IEEE Trans. on Robotics 27 (5) (2011) 943–957.

[19] A. J. Ijspeert, J. Nakanishi, S. Schaal, Trajectory formation for imitation with nonlinear dynamical systems, in: Proc. IEEE Intl Conf. on Intelligent Robots and Systems (IROS), 2001, pp. 752–757.

[20] S. Schaal, P. Mohajerian, A. J. Ijspeert, Dynamics systems vs. optimal control: a unifying view, Progress in Brain Research 165 (2007) 425–445.

[21] H. Hoffmann, P. Pastor, D. H. Park, S. Schaal, Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance, in: Proc. IEEE Intl Conf. on Robotics and Automation (ICRA), 2009, pp. 2587–2592.

[22] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. on Systems, Man, and Cybernetics 15 (1) (1985) 116–132.

[23] F. A. Mussa-Ivaldi, From basis functions to basis fields: vector field approximation from sparse data, Biological Cybernetics 67 (6) (1992) 479–489.

[24] S. Calinon, F. D'halluin, D. G. Caldwell, A. G. Billard, Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework, in: Proc. IEEE-RAS Intl Conf. on Humanoid Robots (Humanoids), Paris, France, 2009, pp. 582–588.

[25] G. Ganesh, A. Albu-Schaffer, M. Haruno, M. Kawato, E. Burdet, Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks, in: Proc. Intl Conf. on Robotics and Automation, 2010, pp. 2705–2711.

[26] T. Flash, N. Hogan, The coordination of the arm movements: an experimentally confirmed mathematical model, Neurology 5 (7) (1985) 1688–1703.

[27] E. Todorov, M. I. Jordan, Optimal feedback control as a theory of motor coordination, Nature Neuroscience 5 (2002) 1226–1235.

[28] M. L. Latash, J. P. Scholz, G. Schoener, Motor control strategies revealed in the structure of motor variability, Exerc. Sport Sci. Rev. 30 (1) (2002) 26–31.

[29] R. Huys, A. Daffertshofer, P. J. Beek, The evolution of coordination during skill acquisition: the dynamical systems approach, in: A. M. Williams, N. J. Hodges (Eds.), Skill Acquisition in Sport: Research, Theory and Practice, Routledge, 2004, pp. 351–373.

[30] M. Bernikera, A. Jarcb, E. Bizzic, M. C. Trescha, Simplified and effective motor control based on muscle synergies to exploit musculoskeletal dynamics, in: Proc. Natl Acad. Sci. USA, Vol. 106, 2009, pp. 7601–7606.

[31] T. Flash, The control of hand equilibrium trajectories in multi-joint arm movements, Biol. Cybern. 57 (4-5) (1987) 257–274.

[32] C. A. Avizzano, E. Ruffaldi, M. Bergamasco, Digital representation of skills for human-robot interaction, in: Proc. IEEE Intl Symp. on Robot and Human Interactive Communication (RO-MAN), 2009, pp. 769 –774.

[33] M. T. Rosenstein, A. G. Barto, R. E. A. Van Emmerik, Learning at the level of synergies for a robot weightlifter, Robotics and Autonomous Systems 54 (8) (2006) 706–717.

[34] D. Pardo, C. Angulo, S. del Moral, A. Catalí, Emerging motor behaviors: Learning joint coordination in articulated mobile robots, Neurocomputing 72 (2009) 3624–3630.

[35] B. Kim, J. Park, S. Park, S. Kang, Impedance learning for robotic contact tasks using natural actor-critic algorithm, IEEE Trans. on Systems, Man, and Cybernetics, Part B 40 (2) (2010) 433–443.

[36] S. Calinon, I. Sardellitti, D. G. Caldwell, Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies, in: Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 2010, pp. 249–254.

[37] P. Kormushev, S. Calinon, D. G. Caldwell, Robot motor skill coordination with EM-based reinforcement learning, in: Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 2010, pp. 3232–3237.

[38] R. Featherstone, D. E. Orin, Robotics foundations: Dynamics, in: B. Siciliano, O. O. Khatib (Eds.), Handbook of Robotics, Springer, Secaucus, NJ, USA, 2008, pp. 35–65.

[39] D. Sternad, S.-W. Park, H. Mueller, N. Hogan, Coordinate dependence of variability analysis, PLoS Computational Biology 6 (4) (2010) 1–16.

[40] N. Bernstein, The co-ordination and regulation of movements, Oxford, UK: Pergamon Press, 1967.

[41] J. Kodl, G. Ganesh, E. Burdet, The CNS stochastically selects motor plan utilizing extrinsic and intrinsic representations, PLoS ONE 6 (9) (2011) 1–10.

[42] A. I. T. Salo, I. N. Bezodis, A. M. Batterham, D. G. Kerwin, Elite sprinting: are athletes individually step-frequency or step-length reliant?, Medicine and Science in Sports and Exercise 43 (6) (2011) 1055–1062.

[43] M. Kobilarov, Cross-entropy motion planning, Intl Journal of Robotics Research 31 (7) (2012) 855–871.

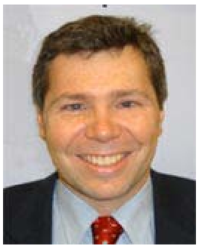
[44] Z. Zhang, C. Chen, J. Sun, K. L. Chan, EM algorithms for Gaussian mixtures with split-and-merge operation, Pattern Recognition 36 (9) (2003) 1973–1983.

[45] H. Mueller, D. Sternad, Decomposition of variability in the execution of goal-oriented tasks: Three components of skill improvement, Journal of Experimental Psychology: Human Perception and Performance 30 (1) (2004) 212–233.

[46] A. L. Thomaz, C. Breazeal, Teachable robots: Understanding human teaching behavior to build more effective robot learners, Artificial Intelligence 172 (2008) 716–737.

Sylvain Calinon is a Team Leader at the Department of Advanced Robotics, Italian Institute of Technology (IIT), where he leads the Learning and Interaction Lab since 2009. He is also a visiting researcher at the Learning Algorithms and Systems Laboratory (LASA), Swiss Federal Institute of Technology in Lausanne (EPFL). He received a PhD on robot programming by demonstration in 2007 from LASA, EPFL, which was awarded by the international Robotdalen scientific award, ABB award and EPFL-Press distinction. From 2007 to 2009, he was a postdoctoral research fellow at LASA, EPFL. His research interests cover robot learning by imitation, machine learning and human-robot interaction.

Petar Kormushev is a Team Leader at the Department of Advanced Robotics, Italian Institute of Technology (IIT). His research interests include robotics and machine learning, especially reinforcement learning. He received PhD in Computational Intelligence from Tokyo Institute of Technology (TiTech). He holds two MSc degrees in Artificial Intelligence and Bio- and Medical Informatics, and a BSc degree in Computer Science from Sofia University. He participated in the European INFRAWEBS project for designing the future Semantic Web, and a Japanese NEDO project for developing common basis for next-generation robots. He received a John Atanasoff award, a St. Kliment Ohridski award, and a 4-year Monbukagakusho/MEXT Japanese research fellowship.



Darwin G. Caldwell is a Director at the Italian Institute of Technology in Genoa, Italy, and a Honorary Professor at the Universities of Sheffield, Manchester, Bangor, Kings College, London and Tianjin University, China. His research interests include innovative actuators, humanoid and quadrupedal robotics and locomotion (iCub, HyQ and COMAN), haptics, exoskeletons, dexterous manipulators, rehabilitation and surgical robotics. He is the author or co-author of over 350 academic papers, and 15 patents and he has received awards from several international journals and conferences. He is an associate editor for the IEEE Trans. on Mechatronics and on the editorial board of the international Journal of Social Robotics and Industrial Robot.