Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

A particle-based policy for the optimal control of Markov decision processes [★]

M. Pirotta [*] **G. Manganini** [*] **L. Piroddi** [*] **M. Prandini** [*] **M. Restelli** [*]

[*] *Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy. E-mail:* {matteo.pirotta, giorgio.manganini, luigi.piroddi}@polimi.it, {prandini, restelli}@elet.polimi.it

Abstract: When the state dimension is large, classical approximate dynamic programming techniques may become computationally unfeasible, since the complexity of the algorithm grows exponentially with the state space size (curse of dimensionality). Policy search techniques are able to overcome this problem because, instead of estimating the value function over the entire state space, they search for the optimal control policy in a restricted parameterized policy space. This paper presents a new policy parametrization that exploits a single point (particle) to represent an entire region of the state space and can be tuned through a recently introduced policy gradient method with parameter-based exploration. Experiments demonstrate the superior performance of the proposed approach in high dimensional environments.

Keywords: Markov decision processes, Stochastic optimal control, Approximate dynamic programming, Reinforcement learning, Policy search.

1. INTRODUCTION

Stochastic optimal control problems arise in several application contexts such as communication networks, manufacturing systems, air traffic management, and power networks. Such problems are particularly challenging due to the stochastic hybrid dynamics of the system, which involves a tight coupling of continuous dynamics, discrete dynamics, and uncertainty. In particular, the computational complexity of the problem increases rapidly with the system size, motivating the quest for efficient algorithms.
In this paper, we investigate the problem of designing an optimal control policy for large scale stochastic systems and determining an approximate solution through a scalable computational procedure. We consider systems that can be modeled as discrete time Markov Decision Processes (MDPs) (Puterman, 1994) with a continuous state space component, and focus on the case where the control space is discrete. The assumption of dealing with MDPs is not restrictive, since they include quite general classes of systems such as discrete time stochastic hybrid systems (Abate et al., 2008).
An MDP is a probabilistic dynamic model where the state evolution is governed by transition probabilities that depend on the control input. If the objective is to maximize some additive reward function along a future time horizon, the optimal control policy that maps each state into the appropriate control action can be characterized through the Dynamic Programming (DP) approach. The rationale of the DP control design methodology is to take into account the immediate impact of the decisions taken at each stage (through some instantaneous reward) as well as their expected future impact over the residual look-ahead time horizon. The maximum expected return that can be obtained starting from any given state is expressed by the optimal value function. If the horizon is infinite and the rewards are discounted, the optimal value function can be characterized as a fixed point of the Bellman optimality equation (Sutton and Barto, 1998). For any MDP, there always exists at least one optimal policy that is a *deterministic*, *i.e.*, where the state-control input map is deterministic.
Unfortunately, the fixed point of the Bellman equation cannot be determined analytically, neither can it be finitely represented, due to the continuous component of the state of the system, and computing the policy calls for some Approximate Dynamic Programming (ADP) method (Sutton and Barto, 1998; Powell, 2007; Busoniu et al., 2010). ADP methods typically consist in an iterative scheme for successively improving the quality of the approximation of the optimal value function (value iteration methods) or of the optimal control policy itself (policy search methods). Parametric approximate representations of the value function or of the control policy are typically introduced to solve the well-known curse of dimensionality afflicting DP. In particular, Reinforcement Learning (RL) approaches (Sutton and Barto, 1998) approximate the expected values involved in the value function computation and policy evaluation using empirical averages over data obtained through the direct interaction between the learning algorithm and the system to be controlled. Such approaches allow to learn optimal or near–optimal policies, even when the system dynamics are unknown or in high dimensional problems.
Policy search methods are among the most effective learning algorithms for dynamical stochastic tasks with con-

[★] Research supported by the European Commission under the MoVeS project, FP7-ICT-2009-257005.

tinuous state and action spaces, and they have been successfully applied to several real tasks (Peters et al., 2003). These methods exploit optimization techniques to search for an optimal policy that maximizes the expected return over some parameterized policy space. In particular, policy gradient methods perform at each iteration a local search in the policy space guided by the gradient of the expected return, that can be estimated from the system trajectories. Unfortunately, standard gradient–based methods, such as REINFORCE or G(PO)MDP (Peters and Schaal, 2008), need to search in the space of stochastic policies, thus suffering from high variance in the gradient estimates, which in turn leads to slow convergence rates.
Recently, a parameter–based exploration strategy (Sehnke et al., 2010) named policy gradient with parameter–based exploration (PGPE) has been presented, that allows to consider *deterministic* policies and introduces a probability distribution over the policy parametrization instead. Then, it searches for the optimal deterministic policy by updating the parameters of such a distribution (denoted hyperparameters) following the estimated gradient direction. In this way, the search in the policy space is replaced by a direct search in the hyperparameter space. As all classical policy gradient approaches, the PGPE estimates the gradient via simulation of system trajectories. However, by moving the stochasticity from the policy to the hyperparameters distribution, it produces low variance trajectories that yield low variance gradient estimates (Zhao et al., 2012). An additional feature of the PGPE is that – differently from standard policy-gradient approaches – it allows the use of policies that are non-differentiable in their parametrization, since the gradient of the expected performance is estimated with respect to the hyperparameters.
In this paper, we propose a novel parametrization of the policy that maps the continuous state space of an MDP onto its finite control space. More precisely, the state feedback policy is represented through particles positioned over the state space and labeled with different control actions. A set of particles defines a policy by partitioning the state space through the associated Voronoi diagram, so that in each state the policy chooses the action associated to the closest particle. Each particle is positioned randomly based on a multivariate Gaussian density function, and a categorical distribution is used to extract the action associated to it. The mean vector and covariance matrix of the Gaussian density function, together with the parameters of the categorical distribution, constitute the hyperparameters to be tuned.

2. PRELIMINARIES

In this section we briefly recall the concept of discrete–time MDP. An MDP is a tuple $\langle X, U, f, r, \gamma, D\rangle$, where X is the state space, U is the action space, $f : X \times X \times U \to \mathbb{R}_+$ is the Markovian transition model where $f(x'|x, u)$ denotes the transition density between state x and state x' under u, $r : X \times U \times X \to \mathbb{R}$ is the reward function, such that $r(x, u, x')$ is the instantaneous reward obtained starting from state x taking action u and reaching state x', $\gamma \in [0, 1)$ is a discount factor, and D is the distribution of the initial state.
A stochastic policy is defined by a probability density function $\pi(\cdot|x)$ over the action space U, which represents the probability of taking action u in state x. When the policy is deterministic, the probability density function becomes a probability mass function concentrated in a single action for each state value. In that case, with a slight abuse of notation, we use π to denote the map between states and actions, *i.e.*, $\pi : X \to U$. We consider infinite horizon problems where the future rewards are exponentially discounted with γ. The value of state x under policy π is expressed as the expected return when starting in x and following π thereafter [1] :

$$V^\pi(x) = \mathbb{E}_{\substack{u(k)\,\sim\,\pi \\ x(k)\,\sim\,f}}\Big[\sum_{k=0}^{\infty}\gamma^k r(x(k), u(k), x(k{+}1))|x(0) = x\Big].$$

Given the initial state distribution D, the policy performance can be evaluated through its expected discounted return:

$$J_D^\pi = \int_X D(x)V^\pi(x)\mathrm{d}x.$$

Solving an MDP implies finding a policy π^* that maximizes the expected reward: $\pi^* \in \arg\max_{\pi\in\Pi} J_D^\pi$, where Π is the set of all (stochastic and deterministic) policies. Interestingly, for any MDP there exists at least one deterministic optimal policy that maximizes $V^\pi(x)$, for all $x \in X$ (Puterman, 1994).
In the following, we consider the problem of finding a policy that maximizes the expected discounted reward over a class of parameterized policies $\Pi_{\boldsymbol{\theta}} = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \mathbb{R}^d\}$, where $\pi_{\boldsymbol{\theta}}$ is a compact notation for $\pi(u|x, \boldsymbol{\theta})$. For ease of notation, we will denote with $J(\boldsymbol{\theta})$ the expected discounted reward of policy $\pi_{\boldsymbol{\theta}}$. In particular, we focus on MDPs with continuous state space $X \subseteq \mathbb{R}^n$ and discrete action space U, with $|U| = m$.

3. THE PGPE ALGORITHM

In this section we describe the policy gradient approach. To this aim we need first to introduce some further notation. Let $h = [x_0, u_0, \ldots, x_T]$ denote a sequence of values for states and actions of finite length T (called history). Given a history h, we define the discounted cumulative reward along h as:

$$r(h) = \sum_{k=0}^{T-1}\gamma^k r(x_k, u_k, x_{k+1}).$$

Then, $J(\boldsymbol{\theta})$ can be approximated as follows

$$J(\boldsymbol{\theta}) \approx \int_H \mathbb{P}\left(h|\boldsymbol{\theta}\right) r(h)\mathrm{d}h,$$

where H is the set of all histories h, and

$$\mathbb{P}(h|\boldsymbol{\theta}) = D(x_0)\prod_{k=0}^{T-1} f(x_{k+1}|x_k, u_k)\pi(u_k|x_k, \boldsymbol{\theta}).$$

Policy gradient approaches update the policy following the direction of the gradient of the expected discounted reward w.r.t. the policy parameters:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx \int_H \mathbb{P}(h|\boldsymbol{\theta})\sum_{k=0}^{T-1}\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(u_k|x_k)r(h)\mathrm{d}h.$$

Since integrating over all the possible histories is practically unfeasible, the previous gradient can only be esti-

[1] The time dependence of stochastic variables is indicated in brackets, *e.g.*, $x(k)$, whereas x_k denotes a possible extracted value of $x(k)$.

mated, *e.g.*, by means of Monte Carlo simulations as in the REINFORCE algorithm (Williams, 1992):

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{k=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(u_k^{(n)} | x_k^{(n)}) r(h^{(n)}),$$

where the histories $h^{(i)}$ are extracted at random according to $\mathbb{P}(h^{(i)}|\boldsymbol{\theta})$. The Monte Carlo estimate of the gradient suffers from high variance (Peters and Schaal, 2008) due to the stochasticity introduced at every transition step by the stochastic policy, which involves sampling the action at every step and, hence, produces several different histories. The PGPE algorithm (Sehnke et al., 2010) is able to reduce the variance problem by replacing the stochastic policy with multiple deterministic policies whose parameters $\boldsymbol{\theta}$ are drawn from a probability distribution $\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho})$ that is a function of a hyperparameter vector $\boldsymbol{\rho}$. PGPE updates $\boldsymbol{\rho}$ following a gradient ascent approach, in order to increase the probability of drawing deterministic policies with higher expected returns. Its main advantage comes from the low variance of the expected returns of histories produced by deterministic policies: the stochasticity is moved to a higher level with the goal of guiding the exploration of the policy parameters $\boldsymbol{\theta}$. Indeed, Zhao et al. (2012) have shown that the gradient estimate in the PGPE has a lower variance than in REINFORCE. Another advantage of the PGPE is that no assumption on the differentiability of the policy w.r.t. to the parameters $\boldsymbol{\theta}$ is required.

The performance measure in the PGPE is the expected value of $J(\boldsymbol{\theta})$ with respect to the distribution $\mathbb{P}(\cdot|\boldsymbol{\rho})$:

$$J(\boldsymbol{\rho}) = \mathbb{E}_{\boldsymbol{\theta} \sim \mathbb{P}(\cdot|\boldsymbol{\rho})}[J(\boldsymbol{\theta})].$$

The goal is to find the best parametrization, *i.e.*, the hyperparameters $\boldsymbol{\rho}^* \in \arg\max_{\boldsymbol{\rho} \in P} J(\boldsymbol{\rho})$. According to the gradient–based optimization:

$$J(\boldsymbol{\rho}) \approx \int_{\Theta} \int_{H} \mathbb{P}(h|\boldsymbol{\theta}) \, \mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho}) \, r(h) \mathrm{d}h \mathrm{d}\boldsymbol{\theta},$$

and the hyperparameters $\boldsymbol{\rho}$ are updated following the gradient direction:

$$\boldsymbol{\rho}_{i+1} = \boldsymbol{\rho}_i + \beta_i \nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}_i),$$

where, using the equivalence $(\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho}) \nabla_{\boldsymbol{\rho}} \log \mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho}) = \nabla_{\boldsymbol{\rho}} \mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho}))$, we can write:

$$\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}) = \int_{\Theta} \mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho}) \nabla_{\boldsymbol{\rho}} \log \mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho}) \int_{H} \mathbb{P}(h|\boldsymbol{\theta}) r(h) \mathrm{d}h \mathrm{d}\boldsymbol{\theta},$$

The previous equation cannot be solved directly because the integration over the entire space of histories and policy parameters is unfeasible. However, a sampling method can be exploited for the gradient estimation:

$$\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}) \approx \frac{1}{N} \sum_{n=1}^{N} r(h^{(n)}) \nabla_{\boldsymbol{\rho}} \log \mathbb{P}(\boldsymbol{\theta}^{(n)}|\boldsymbol{\rho}).$$

where the pairs $(\boldsymbol{\theta}^{(n)}, h^{(n)})$ are extracted independently according to the following mechanism: parameter $\boldsymbol{\theta}^{(n)}$ is drawn first from $\mathbb{P}(\cdot|\boldsymbol{\rho})$ and then history $h^{(n)}$ is drawn from the conditional distribution $\mathbb{P}(\cdot|\boldsymbol{\theta}^{(n)})$. As a consequence, the PGPE implementation requires to generate at every iteration a dataset $\mathcal{D} = \{\boldsymbol{\theta}^{(n)}, h^{(n)}\}_{n=1}^{N}$.

To further reduce the variance of the gradient estimation, a baseline $b_{\boldsymbol{\rho}}$ can be introduced (Zhao et al., 2012):

$$\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}) \approx \frac{1}{N} \sum_{n=1}^{N} (r(h^{(n)}) - b_{\boldsymbol{\rho}}) \nabla_{\boldsymbol{\rho}} \log \mathbb{P}(\boldsymbol{\theta}^{(n)}|\boldsymbol{\rho}).$$

4. POLICY PARAMETRIZATION VIA PARTICLES

To the best of the authors' knowledge, the PGPE method has only been applied to policy parametrizations that are linear in the parameters: $\pi_{\boldsymbol{\theta}}(x) = \phi(x)^{\mathrm{T}} \boldsymbol{\theta}$, where $\boldsymbol{\theta}$ are drawn from a multivariate Gaussian distribution. Such parametrizations have been shown to be effective in high dimensional continuous problems. However, they are not directly applicable in the case when the action space is a finite discrete set. In this section, we present a new parametrization of a policy with discrete actions, and we incorporate the parametrization into the PGPE framework. The idea behind the adopted parametrization is inspired by clustering. We use *particles* labeled with actions to identify the regions of the state space X that are mapped to different control actions. More precisely, a particle is a point in the state space X with a label in U, and the policy deterministically associates to $x \in X$ the action defined by the label of the particle that is closest to x. By increasing the number of particles, one can in principle reproduce the map associated with the optimal policy with arbitrary accuracy.

More formally, $\pi_{\boldsymbol{\theta}}$ is a deterministic policy described by a set of p labeled particles $\boldsymbol{\theta} = \{\boldsymbol{\theta}^{(i)}\}_{i=1}^{p}$ where each particle $\boldsymbol{\theta}^{(i)} = \left[x^{(i)}, u^{(i)}\right]^{\mathrm{T}}$ belongs to the joint space $X \times U$ [2]. Given $x \in X$, the action u associated to x by policy $\pi_{\boldsymbol{\theta}}$ is obtained via the k–NN (k–nearest neighbors) algorithm with k equal to 1:

$$\pi_{\boldsymbol{\theta}}(x) = \left\{u^{(i)} \middle| \Delta(x, x^{(i)}) \leq \Delta(x, x^{(j)}), \forall j \neq i\right\},$$

where $\Delta\left(x, x^{(i)}\right)$ is the Euclidean distance between state x and the state component $x^{(i)}$ of the i–th particle. As a result, the state space is partitioned in polyhedral sets as shown in Figure 1.

A basic strategy consists in selecting a sufficiently high number of particles, each one with an associated *fixed* action, and learn the optimal position of each particle. However, this approach can be very inefficient in complex problems where the number of particles required by each action may be different and *a priori* unknown. A better approach, that overcomes this "preallocation" problem, is to exploit a learning process both for positioning the particle over the state space and determining the associated action.

According to the PGPE scheme, the policy parameters are drawn from some joint distribution $\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho})$. The p particles $\boldsymbol{\theta}^{(i)}$ are independent, and each one has an independent distribution over the state and action components, that is $x^{(i)} \sim \mathcal{N}(\cdot; \mu^{(i)}, \Sigma^{(i)})$ and $u^{(i)} \sim \mathcal{B}(\cdot; \alpha^{(i)})$, where:

$$\mathcal{B}(u_l; \alpha^{(i)}) = \frac{e^{\alpha_l^{(i)}}}{\sum_{j=1}^{m} e^{\alpha_j^{(i)}}},$$

is a Boltzmann distribution (u_l denotes the l–th action in U). The hyperparameter vector $\boldsymbol{\rho}$ is then defined as follows: $\boldsymbol{\rho} = [\mu^{(i)}, \Sigma^{(i)}, \alpha^{(i)}]_{i=1}^{p}$.

A graphical representation of the hyperparameters distribution $\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho})$ is given in Figure 1, together with a policy

[2] From now on, superscript (i) denotes the i–th particle.

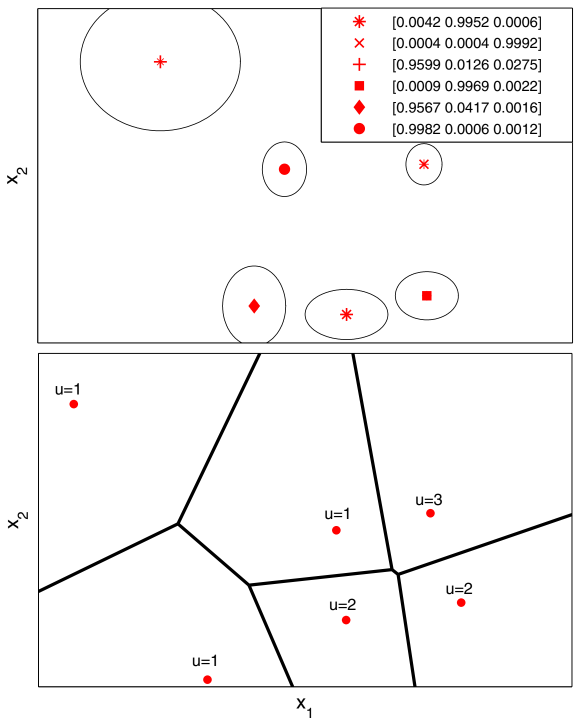

Fig. 1. Illustrative picture of a policy parametrization and of an extracted policy in a two dimensional state space ($X \subseteq \mathbb{R}^2$) with 3 actions ($|U| = 3$). Top: Gaussian distributions of 6 particles (the red symbols and the ellipses represent the mean values and the standard deviation isolevel curves, respectively), with the associated action probabilities. Bottom: Voronoi diagram of a deterministic policy extracted from the distributions of the particles.

with parameters extracted from $\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho})$.
In order to reduce the number of parameters, we discard the cross-correlation terms in the covariance matrix, so that $\Sigma^{(i)} = diag(\Sigma_{11}^{(i)}, \Sigma_{22}^{(i)}, \ldots, \Sigma_{nn}^{(i)})$. Moreover, in order to prevent the variance from becoming negative we exploit the parametrization presented by Kimura and Kobayashi (1998), where $\Sigma_{jj}^{(i)}$ is represented by a logistic function parameterized by $\sigma_j^{(i)}$: $\Sigma_{jj}^{(i)} = \frac{\tau}{1+e^{-\sigma_j^{(i)}}}$.
The total number of hyperparameters is then $2n \cdot m \cdot p$, where n is the dimension of the state space X, m is the number of actions in U, and p is the number of particles. The partial derivatives of $\log \mathbb{P}(\boldsymbol{\theta}|\boldsymbol{\rho})$ with respect to $\mu^{(i)}$, $\Sigma^{(i)}$ and $\alpha^{(i)}$ are given by:

$$\begin{aligned}
\nabla_{\mu^{(i)}} \log \mathbb{P}(\boldsymbol{\theta}^{(j)}|\boldsymbol{\rho}) =& \delta_{ij} (\Sigma^{(j)})^{-1} (x^{(j)} - \mu^{(j)}) \\
\nabla_{\sigma_l^{(i)}} \log \mathbb{P}(\boldsymbol{\theta}^{(j)}|\boldsymbol{\rho}) =& \delta_{ij} \frac{e^{-\sigma_l^{(i)}}}{2\left(1 + e^{-\sigma_l^{(i)}}\right)} \\
& \times \left[(x^{(j)} - \mu^{(j)})_l^2 (\Sigma_{ll}^{(j)})^{-1} - 1 \right] \\
\nabla_{\alpha_l^{(i)}} \log \mathbb{P}(\boldsymbol{\theta}^{(j)}|\boldsymbol{\rho}) =& \tau \delta_{ij} \left(\delta_{u^{(j)} u_l} - \mathcal{B}(u_l; \alpha^{(j)}) \right),
\end{aligned}$$

where $\delta_{ij} = 1$ when $i = j$, and 0 otherwise.

5. SIMULATION EXAMPLE

We next present the results of a computational study for the multi–room heating benchmark described by Fehnker and Ivančić (2004), and addressed also in (Abate et al., 2007, 2008; Prandini and Piroddi, 2012). The problem concerns the simultaneous temperature regulation in n rooms, assuming that each room has a heater, but that at most one heater at a time can be active. A switching control strategy must be designed that decides at each time step which room should be heated, depending on the temperature values in all the rooms. If the temperature of one or more rooms exits the prescribed range, the control strategy should automatically recover the desired condition in the shortest time possible. A nice feature of this benchmark is that it is suitable for testing the scalability of the proposed policy gradient approach, since the problem dimensionality can be easily increased by adding more rooms.

5.1 Model of the multi-room heating system

The multi-room heating system can be modeled as a stochastic hybrid system with hybrid state $s = (q, x)$, where the discrete state component q identifies the actual room being heated, while the continuous state $x = (x_1, \ldots, x_n) \in X = \mathbb{R}^n$ represents the (average) temperature in each room. Accordingly, the discrete state space is defined as $\mathcal{Q} = \{1, \ldots, n+1\}$, where in mode $q = i$, $i = 1, \ldots, n$, the i–th room is heated, while no room is heated when $q = n + 1$. The control space is given by $U = \{1, \ldots, n+1\}$, where $u = i$, $i = 1, \ldots, n$, corresponds to the command of heating the i–th room, while $u = n+1$ is the command to switch the heater off. The average temperature in room i is ruled by the following stochastic difference equation, obtained by Euler discretization of the corresponding continuous time dynamics with constant time step Δt:

$$\begin{aligned}
x_i(k+1) = x_i(k) + b_i(x_a - x_i(k))\Delta t + c_i h_i(k) \Delta t \qquad (1) \\
+ \sum_{j=1,\ldots,n; j \neq i} a_{ij}(x_j(k) - x_i(k))\Delta t + n_i(k), \; i = 1, \ldots, n,
\end{aligned}$$

where $x_i(k)$ is the average temperature in room i at time k, x_a is the ambient temperature (assumed constant), and $h_i(k)$ is a boolean function equal to 1 if $q(k) = i$ (*i.e.*, when room i is heated), and 0 otherwise. Parameters a_{ij}, b_i and c_i in (1) are non-negative constants representing the heat exchange coefficients between room i and room j (a_{ij}), the heat loss rate of room i to the ambient (b_i) and the heat rate supplied by the heater in room i (c_i), all normalized with respect to the average thermal capacity of room i. Finally, the disturbance $n_i(k)$ affecting the temperature of room i is assumed to be a sequence of i.i.d. Gaussian random variables with zero mean and variance $\nu^2 \Delta t$, independent of $n_j(k)$, $j \neq i$.
The heaters are controlled by a thermostat that is prone to delay and switching failures. This is modeled through a discrete transition probability function that governs the mode transitions:

$$f_q(q'|q,u) = \begin{cases} 1, & u = q = q' \\ 1-\alpha, & u \neq q, q' = q \\ \alpha, & u = q', q' \neq q \end{cases}, \qquad (2)$$

where $\alpha \in [0, 1]$ is the one-step delay/failure probability.
The desired operating region is given by

$$A = \{(x_1, \ldots, x_n) : \; x_i \in [x_l, x_u], \; i = 1, \ldots, n\},$$

where x_l and x_u specify the lower and upper bounds for the temperature in each room.

5.2 *Control strategy*

In order to design the switching control strategy for the multi–room heating system we adopt the self-recovery approach proposed by Prandini and Piroddi (2012) where the objective is to keep the state of the system within A and drive it back to A as soon as possible in case of exit. The control design problem can be formulated as in Section 2, where the heater is off in the initial state and the reward function $r : X \times X \to \{0, \pm 1, \pm 2, \dots, \pm n\}$ is defined as:

$$r(x, x') = g(x') - g(x)$$

where $g(x) = \sum_{i=1}^{n} \mathbf{1}_{[x_l, x_u]}(x_i)$ is the number of rooms whose temperature is within the desired range $[x_l, x_u]$. This means that transitions leading the temperature of one room outside $[x_l, x_u]$ are penalized with -1, transitions leading the temperature of one room back into $[x_l, x_u]$ are rewarded with $+1$, whereas all other transitions do not provide neither a penalty nor a reward. The discount factor γ re-scales penalties and rewards with time, encouraging early entrances in and late exits from A, and overall reducing the duration of periods outside A (Prandini and Piroddi, 2012).

5.3 *Numerical results*

The proposed approach has been tested for the n–room case with $n = 1, \dots, 5$ with two different objectives. First, a *safety problem* has been addressed, optimizing the control policy starting from the interior of the safe zone A (all rooms are initially at 19 °C), with the objective of remaining inside set A as much as possible. Then, a *recovery problem* has been solved, where the starting point of the optimization procedure is placed outside A (all rooms are initially at 17 °C), with the objective of reaching the safe set A as soon as possible, and remaining inside thereof.
The following parameters have been employed in the experiments: $\Delta t = 1/30$, $\nu = 1$, $x_a = 6$, $b_i = 0.25$ and $c_i = 12$ for $i = 1, \dots, 5$, $a_{ij} = a_{ji} = 0.33$, for $i = 1, \dots, 4$ with $j = i+1$, $\alpha = 0.8$. As for the safe temperature range, we chose $[x_l, x_u] = [17.5, 22]$. A discount factor $\gamma = 0.999$ was used in the calculation of the total reward. The time horizon length has been set to $T = 100$. The learning rates are set equal to the variance for the Gaussian parameters and to 1 for the Boltzmann parameters.
The same problem with $n \leq 3$ has been addressed in (Prandini and Piroddi, 2012) based on an ADP scheme with continuous state gridding. Based on those results, where very similar policies resulted for different values of the discrete mode, the latter has not been considered in the policy parametrization (which depends only on the continuous state components), although it enters the dynamics of the system.
To illustrate the results we first make reference to the 2–room case. The PGPE algorithm has been applied using 3 particles only (each associated to a specific control action). Given the simplicity of the underlying control problem, increasing the number of particles has not provided significant improvements. The mean values of the particles have been initialized randomly, while the initial variance has been set to 9. A maximum of 250 iterations was allowed to achieve convergence, which is ascertained through a condition of the type $\|\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho})\| < \epsilon$, where

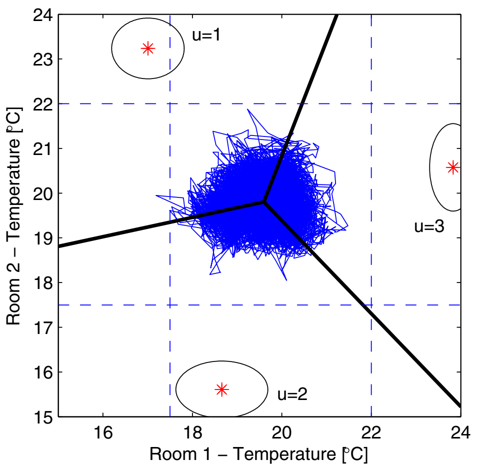

Fig. 2. Safety problem. The Voronoi diagram defining the policy associated to the learned parameter vector $\boldsymbol{\rho}$ is depicted considering the mean position $\mu^{(i)}$ of particles. The ellipses represent the isolevel curves associated to the standard deviations of the Gaussian distributions of the particles. The labels represent the deterministic actions $u^{(i)}$. 200 trajectories obtained following the mentioned policy starting from $x(0) = [19\ 19]^T$ are reported (blue lines).

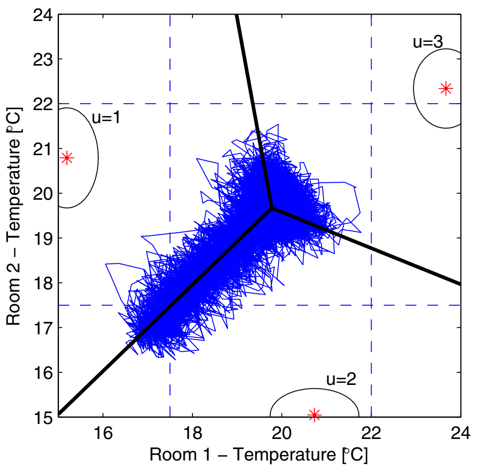

Fig. 3. Recovery problem. Optimal policy parametrization and 200 trajectories starting from $x(0) = [17\ 17]^T$.

$\epsilon = 0.001$ is a suitably small threshold (typically, for the 2–room case convergence is achieved much earlier than the allotted iterations). At each iteration the gradient is evaluated based on 1000 deterministic policies drawn from the current hyperparameters distribution. Each policy is evaluated on a sample trajectory of length 100. Figure 2 reports the hyperparameters of the policy distribution and the deterministic policy corresponding to the mean values of the particles. The figure also shows 200 trajectories starting from $x(0) = [19\ 19]^T$. All trajectories remain inside A on average 99.99% of the time (only two isolated samples are outside A). The policy has been further validated testing different initial conditions drawn uniformly inside A, resulting in a 99.77% permanence inside A on average.
The recovery problem has been also addressed with reference to the 2–room case. Figure 3 illustrates the hyperparameters, the mean policy and a set of trajectories starting from $x(0) = [17\ 17]^T$. This second problem results in a

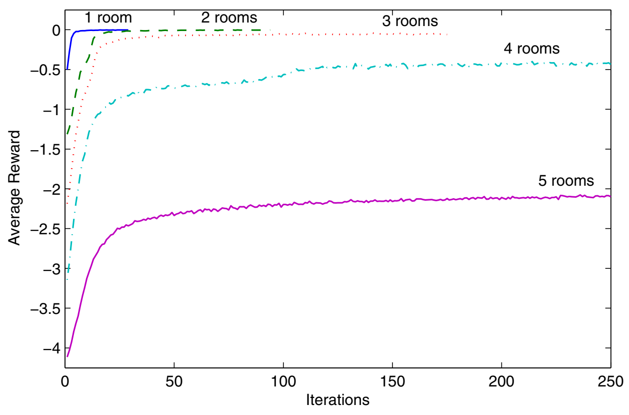

Fig. 4. Safety problem ($n = 1, \dots, 5$): reward evolution as a function of the algorithm iterations.

Table 1. Computational results for the safety problem: PGPE algorithm.

Number of rooms	Particles	Iterations	CPU time [s]	Memory [kB]
1	2	29	362	6
2	3	94	1000	33
3	4	175	1500	70
4	5	250	1780	116
5	6	250	1788	162

Table 2. Computational results for the safety problem: gridding approach.

Number of rooms	Bins	Iterations	CPU time [s]	Memory [kB]
1	30	44	0	20
2	30	36	100	3800
3	30	69	8836	10^6

wider exploration of the state space, and not surprisingly the resulting policy is much more similar to the optimal one according to Prandini and Piroddi (2012). All trajectories of the tested policy terminate inside A. This requires 8.19 steps on average.

Both the safety and recovery problems have been applied to larger case instances, up to 5 rooms. In each case $|U| = n+1$ particles have been employed only, each associated to a specific action, and the same iteration limit and number of policy extractions have been considered. The policy evaluation cost increases linearly with the state vector dimension. Due to the augmented complexity of the optimization problem (the parameter space is larger), a longer time is typically required to achieve convergence (see, *e.g.*, Figure 4). Notice that while the optimal expected reward value ($J(\boldsymbol{\rho}^*) = 0$) is achieved for the low dimensional problems, lower rewards are obtained for $n = 4, 5$. This is justified by the under-actuated nature of the problem (only one room at a time can be heated). Finally, Table 1 reports some figures regarding the computational load of the approach. Both the average computational time and the memory occupancy are included. Notice that for the proposed approach the CPU time and the memory occupancy increase linearly with the problem dimension (the number of particles is chosen equal to $n+1$), as opposed to the previously developed ADP approach based on gridding which scales unfavorably with the state dimension (see Table 2, where the state is gridded uniformly using 30 bins per dimension).

6. CONCLUSIONS

This paper has investigated a policy search technique that applies to stochastic systems with continuous state and discrete action spaces, and appears to scale favorably with state space size. The presented approach is based on the PGPE policy-gradient technique, endowed with a novel policy parametrization using particles to describe entire areas of the state space associated to the same action. By encapsulating the policy parametrization in the PGPE framework, it is possible to automatically learn both the position of the particle in the state space and the associated action. This ability comes at the price of a high number of parameters to be tuned, that scales proportionally to the number of particles. However, the number of samples required by the algorithm in order to estimate the gradient direction is not strictly related to the number of parameters (*i.e.*, particles) and does not increase exponentially with the state dimension.

REFERENCES

Abate, A., Amin, S., Prandini, M., Lygeros, J., and Sastry, S. (2007). Computational approaches to reachability analysis of stochastic hybrid systems. In A. Bemporad, A. Bicchi, and G. Buttazzo (eds.), *Hybrid Systems: Computation and Control*, number 4416 in Lecture Notes in Computer Sciences, 4–17. Springer-Verlag, Berlin.

Abate, A., Prandini, M., Lygeros, J., and Sastry, S. (2008). Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44(11), 2724–2734.

Busoniu, L., Babuska, R., Schutter, B.D., and Ernst, D. (2010). *Reinforcement Learning and Dynamic Programming Using Function Approximators.* CRC Press, Inc., Boca Raton, FL, USA, 1st edition.

Fehnker, A. and Ivančić, F. (2004). Benchmarks for hybrid systems verifications. In R. Alur and G. Pappas (eds.), *Hybrid Systems: Computation and Control*, LNCS 2993, 326–341. Springer Verlag.

Kimura, H. and Kobayashi, S. (1998). Reinforcement learning for continuous action using stochastic gradient ascent. *Intelligent Autonomous Systems (IAS-5)*, 288–295.

Peters, J., Vijayakumar, S., and Schaal, S. (2003). Reinforcement learning for humanoid robotics. In 3rd *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids2003).* Karlsruhe, Germany.

Peters, J. and Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4), 682–697.

Powell, W. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality.* John Wiley & Sons.

Prandini, M. and Piroddi, L. (2012). A self-recovery approach to the probabilistic invariance problem for stochastic hybrid systems. In 51st *IEEE Conference on Decision and Control*, 2096–2102. Maui (HI), USA.

Puterman, M.L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley-Interscience, New York, NY.

Sehnke, F., Osendorfer, C., Rückstieß, T., Graves, A., Peters, J., and Schmidhuber, J. (2010). Parameter-exploring policy gradients. *Neural Networks*, 23(4), 551–559.

Sutton, R.S. and Barto, A.G. (1998). *Reinforcement Learning: An Introduction.* The MIT Press.

Williams, R.J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3–4), 229–256.

Zhao, T., Hachiya, H., Niu, G., and Sugiyama, M. (2012). Analysis and improvement of policy gradient estimation. *Neural Networks*, 26(0), 118–129.