Robust Policy Updates for Stochastic Optimal Control

Elmar Rueckert[1], Max Mindt[1], Jan Peters[1,2] and Gerhard Neumann[3]

***Abstract*— For controlling high-dimensional robots, most stochastic optimal control algorithms use approximations of the system dynamics and of the cost function (e.g., using linearizations and Taylor expansions). These approximations are typically only locally correct, which might cause instabilities in the greedy policy updates, lead to oscillations or the algorithms diverge. To overcome these drawbacks, we add a regularization term to the cost function that punishes large policy update steps in the trajectory optimization procedure. We applied this concept to the Approximate Inference Control method (AICO), where the resulting algorithm guarantees convergence for uninformative initial solutions without complex hand-tuning of learning rates. We evaluated our new algorithm on two simulated robotic platforms. A robot arm with five joints was used for reaching multiple targets while keeping the roll angle constant. On the humanoid robot Nao, we show how complex skills like reaching and balancing can be inferred from desired center of gravity or end effector coordinates.**

I. INTRODUCTION

Typical whole body motor control tasks of humanoids, like reaching for objects while walking, avoiding obstacles during motion, or maintaining balance during movement execution, can be characterized as optimization problems with multiple criteria of optimality or objectives. The objectives may be specified in the robot's configuration space (e.g., joint angles, joint velocities and base reference frame), in task space (where objectives such as desired end effector coordinates or center of gravity positions are specified), or in combinations of both. In this paper, we consider control problems in nonlinear systems with multiple objectives in combinations of these spaces.

A common strategy to whole body motor control is to separate the redundant robot's configuration space into a task space and an orthogonal null space. Objectives or optimality criteria of motion are implemented as weights or priorities [1] to the redundant solutions in the null space. While these approaches have been successfully applied to a variety of tasks, including reaching, obstacle avoidance, walking and maintaining stability [2]–[5], the application of these methods is typically limited to motor control and can not be directly used for motor planning. It is also unclear how these methods can be applied to motor control problems in nonlinear systems like compliant robots.

Alternatively, in Stochastic Optimal Control (SOC) problems [6], a movement policy is optimized with respect to a cost function, which combines the different criteria of optimality with different weightings. For nonlinear systems, SOC methods use approximations of the system dynamics and of the cost functions, e.g., through linearizations and 2nd order Taylor expansions. These approximations are only locally correct and the updates of the policy may become unstable if the minima is not close to the points of the linearizations, or may oscillate in the case of multiple solutions.

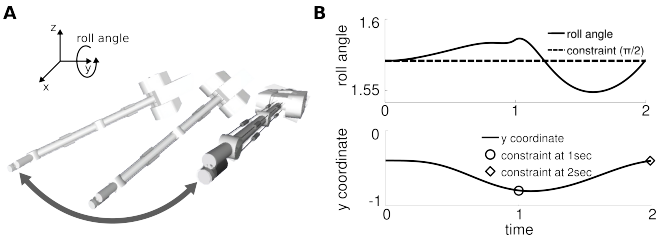

Fig. 1. A 5-degree-of-freedom robot arm has to reach for a via-point (the posture on the left in A) and return to its initial pose (the posture on the right in A). The reaching task is encoded in four task objectives, i.e., three Cartesian coordinates and the roll angle of the end effector. The inferred trajectories for the y coordinate and the roll angle, including the objectives, are shown in (B).

Many SOC methods address this issue and implement regularizations on the *algorithmic level*. E.g., in the *iLQG* method [7] a diagonal regularization term is added to the control cost Hessian[1], and in an extension [8], it was suggested to penalize deviations from the state trajectory used for linearization rather than controls. A drawback of this approach is that the additive regularization term needs rapid re-scaling to prevent divergence and accurate fine-tuning of a learning rate to find good solutions, which is challenging and increases the computational time of the algorithm.

Probabilistic planning methods that translate the SOC problem into an inference problem, typically implement learning rates in their belief updates [9] or in the feedback controller [10]. However, in nonlinear systems, both strategies are suboptimal in the sense that even with a small learning rate on the beliefs the corresponding control updates might be large (and vice-versa, respectively).

We propose to regularize the policy updated on the *cost function level* for probabilistic planning. We also penalize large distances between two successive trajectories in the iterative trajectory optimization procedure. In [8], the regularization term is only used for the control gains and not for the updates of the value function. However, the deviation from the linearization point can still be high if small regularization

[1]Intelligent Autonomous Systems Lab, Technische Universität Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany {rueckert, mindt}@ias.tu-darmstadt.de

[2]Robot Learning Group, Max-Planck Institute for Intelligent Systems, Tuebingen, Germany mail@jan-peters.net

[3]Computational Learning for Autonomous Systems, Hochschulstr. 10, 64289 Darmstadt, Germany neumann@ias.tu-darmstadt.de

[1]The update step in the trajectory optimizer corresponds to a Gauss-Newton Hessian approximation [8].

terms are used. In our approach, we always want to stay close to the linearization point as the used approximations are only locally correct. Hence, using too large update steps by greedily exploiting the inaccurate models might again be dangerous, leading the instabilities or oscillations. The scaling parameter of our punishment term serves as step size of the policy update. Due to the use of probabilistic planning, the need of an additional learning rate and complex update strategies of this learning rate can be avoided. Moreover, we will demonstrate that this type of regularization results in more robust policy updates in comparison to [8]. We choose the Approximate Inference Control (AICO) algorithm as probabilistic planning method [9] to discuss and analyze the proposed regularization, however, the same "*trick*" can be applied to large variety of SOC methods.

In the rest of this paper, we introduce the probabilistic planning method AICO, analyze its convergence properties in a reaching task in a light-weight robot arm and introduce the proposed regularization on the *cost function level*. The resulting algorithm is evaluated on the humanoid robot Nao, where in first results, arm reaching and balancing skill are inferred from desired center of gravity or end effector coordinates. We conclude in Section IV.

II. METHODS

We consider finite horizon Markov decision problems[2]. Let $\boldsymbol{q}_t \in \mathbb{Q}$ denote the current robot's state in configuration space (e.g., a concatenation of joint angles, joint velocities and reference coordinates in floating base systems) and let vector $\boldsymbol{x}_t \in \mathbb{X}$ denote task space features like end effector positions or the center of gravity of a humanoid (these features will be used to specify a cost function later). At time t, the robot executes the action $\boldsymbol{u}_t \in \mathbb{U}$ according to the movement policy $\pi(\boldsymbol{u}_t|\boldsymbol{q}_t)$.

The chosen action at the current state is evaluated by the cost function $C_t(\boldsymbol{q}_t, \boldsymbol{u}_t) \in \mathbb{R}^1$ and results in a state transition characterized by the probability $P(\boldsymbol{q}_{t+1}|\boldsymbol{q}_t, \boldsymbol{u}_t)$. In Stochastic Optimal Control (SOC), the goal is to find a stochastic policy π^* that minimizes the expected cost

$$\pi^* = \underset{\pi}{\operatorname{argmin}} \left\langle C_T(\boldsymbol{q}_T) + \sum_{t=0}^{T-1} C_t(\boldsymbol{q}_t, \boldsymbol{u}_t) \right\rangle_{q_\pi} \quad , \qquad (1)$$

where the expectation, denoted by the symbols $\langle\cdot\rangle$, is taken with respect to the trajectory distribution

$$q_\pi(\boldsymbol{q}_{0:T}, \boldsymbol{u}_{0:T-1}) = P(\boldsymbol{q}_0) \prod_{t=0}^{T-1} \pi(\boldsymbol{u}_t|\boldsymbol{q}_t) P(\boldsymbol{q}_{t+1}|\boldsymbol{q}_t, \boldsymbol{u}_t) \ ,$$

where $P(\boldsymbol{q}_0)$ is the initial state distribution.

A. Bayesian inference for control

An interesting class of algorithms to SOC problems have been derived by reformulating the original Bellman formulation in (1) as an Bayesian inference problem [14]–[17].

[2]Note that the same principle of regulating the update steps in trajectory optimization can also be applied to planning algorithms in infinite horizon problems such as [11], [12]

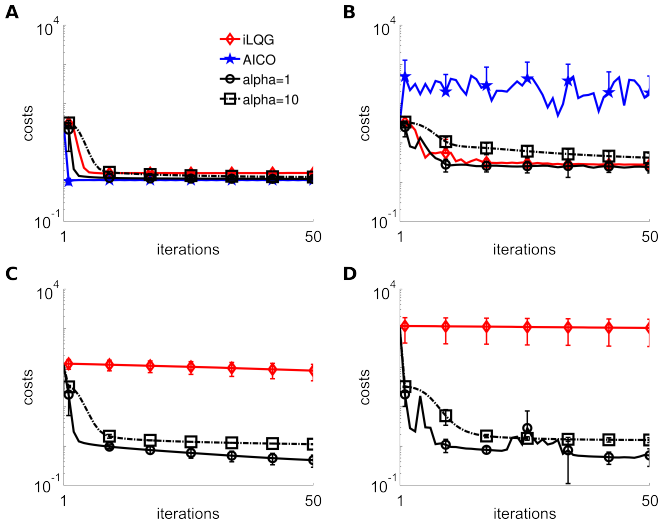

Fig. 2. Comparison of the convergence properties of *iLQG*, AICO and our robust variant, where the rate of convergence is controlled via the parameter α. In the top row (A-B), the model of the forward dynamics was approximated by a *pseudo dynamics* model [13]. In the bottom row, an analytic forward dynamics model of a 5-degree-of-freedom robot arm was used. The panels in the first column denote the costs of the planning algorithms applied to a simple task, where the robot arm has to reach for an end effector target and return to the initial state. In the second column (B,D), the robot has to keep additionally the roll angle constant (at $\pi/2$). Shown are the mean and the standard deviations for 10 initial states '$\boldsymbol{q}_0$ sampled from a Gaussian with zero mean and a standard deviation of 0.05.

Instead of minimizing costs, the idea is to maximize the probability of receiving a reward event ($r_t = 1$) at every time step

$$p(r_t = 1|\boldsymbol{q}_t, \boldsymbol{u}_t) \propto \exp\{-C_t(\boldsymbol{q}_t, \boldsymbol{u}_t)\} \quad . \qquad (2)$$

Note that the idea of turning the cost function in Eq. (1) into a reward signal was also used in operational space control approaches [18], [19].

In the probabilistic framework, we want to compute the posterior over state and control sequences, conditioning on observing a reward at every time step,

$$p_\pi(\boldsymbol{q}_{0:T}, \boldsymbol{u}_{0:T-1}|r_{0:T} = 1) = \exp\{-C_T(\boldsymbol{q}_T)\} \\ q_\pi(\boldsymbol{q}_{0:T}, \boldsymbol{u}_{0:T-1}) \prod_{t=1}^{T-1} p(r_t = 1|\boldsymbol{q}_t, \boldsymbol{u}_t) \quad .$$

For efficient implementations of this inference problem, a number of algorithms have been proposed that apply iterative policy updates assuming that all probability distributions can be modeled by an instance of the family of *exponential* distributions [9], [20], [21]. We will restrict our discussion on the Approximate Inference Control (AICO) algorithm with Gaussians [9].

B. Approximate inference control with Gaussians (AICO)

We consider system dynamics of the form $\boldsymbol{q}_{t+1} = f(\boldsymbol{q}_t, \boldsymbol{u}_t) + \epsilon$ with ϵ denoting zero mean Gaussian noise. In AICO (with Gaussians), the system dynamics are linearized through 1st order Taylor expansions, i.e.,

$P(\boldsymbol{q}_{t+1}|\boldsymbol{q}_t, \boldsymbol{u}_t) = \mathcal{N}(\boldsymbol{q}_{t+1}|A_t\boldsymbol{q}_t + a_t + B_t\boldsymbol{u}_t, Q_t)$, where the state transition matrix A_t, the linear drift term a_t and the control matrix B_t are often computed with derivatives simulated through finite differences methods. The numerical stability of AICO also depends on the accuracy of the linearized model, we will therefore additionally compare to an approximation of the system dynamics, where controls $\boldsymbol{u}_t$ correspond directly to joint accelerations[3]. We will refer to this approximation as *pseudo-dynamic* model.

We propose to add a regularization term to the cost function. Before explaining the regularization term in more detail, we briefly discuss how different objectives are implemented in AICO. In the simplest case, the task-likelihood function in (2) can be split into separate state and a control dependent terms, i.e.,

$$p(r_t = 1|\boldsymbol{q}_t, \boldsymbol{u}_t) = \mathcal{N}[\boldsymbol{q}_t|\boldsymbol{r}_t, R_t]\,\mathcal{N}[\boldsymbol{u}_t|\boldsymbol{h}_t, H_t] \quad , \quad (3)$$

where, for analytical reasons, the Gaussians are given in *canonical* form, i.e., $\mathcal{N}[\boldsymbol{u}_t|\boldsymbol{h}_t, H_t] \propto \exp(-1/2\,\boldsymbol{u}_t^T H_t \boldsymbol{u}_t + \boldsymbol{u}_t^T \boldsymbol{h}_t)$. Note that the vector $\boldsymbol{r}_t$ in (3) denotes the linear term for the Gaussian distribution and must not be confused with the auxiliary variable $r_t = 1$ in (2) denoting a reward event. By inserting (3) in (2) we obtain the quadratic costs,

$$C_t(\boldsymbol{q}_t, \boldsymbol{u}_t) = \boldsymbol{q}_t^T R_t \boldsymbol{q}_t - 2\boldsymbol{r}_t^T \boldsymbol{q}_t + \boldsymbol{u}_t^T H \boldsymbol{u}_t - 2\boldsymbol{h}_t^T \boldsymbol{u}_t \quad . \quad (4)$$

The state dependent costs, encoded by $\mathcal{N}[\boldsymbol{q}_t|\boldsymbol{r}_t, R_t]$, can be defined in configuration space[4], in task space[5], or even in combinations of both spaces [16].

On the algorithmic level, AICO combines forward messages and backward messages to compute the belief over trajectories. We represent these Gaussian forward message by $N[\boldsymbol{q}_t|\boldsymbol{s}_t, S_t]$, the backward message by $N[\boldsymbol{q}_t|\boldsymbol{v}_t, V_t]$, and the belief by $N[\boldsymbol{q}_t|\boldsymbol{b}_t, B_t]$. The recursive update equations are given in [9] and in [10] where an implementation which additionally implements control constraints (otherwise $\boldsymbol{h}_t = 0$) is given.

We can also compute the most likely action given the task constraints. By doing so, in the case of AICO with Gaussians, we obtain a time varying linear feedback controller

$$\boldsymbol{u}_t^{[n]} = \boldsymbol{o}_t + O_t \boldsymbol{q}_t \quad , \quad (5)$$

where $\boldsymbol{o}_t$ is an open loop gain and O_t denotes the feedback gain matrix (n denotes the iteration).

[3]For a single joint with $\boldsymbol{q} = [q, \dot{q}]^T$, the matrix $A = \left(\begin{smallmatrix} 1 & \tau \\ 0 & 1 \end{smallmatrix}\right)$, $a = \left(\begin{smallmatrix} 0 \\ 0 \end{smallmatrix}\right)$, and $B = \left(\begin{smallmatrix} \tau^2 \\ \tau \end{smallmatrix}\right)$, where τ denotes the time step.

[4]Reaching a goal state $\boldsymbol{g}^* \in \mathbb{Q}$ in configuration space can be encoded by $\boldsymbol{r}_t = R_t \boldsymbol{g}^*$ where the precision matrix R_t scales the importance of different dimensions.

[5]Let $\boldsymbol{x}^* \in \mathbb{X}$ denote a desired end effector position and let $\boldsymbol{x} = f(\boldsymbol{q})$ be the forward kinematics mapping and $J(\boldsymbol{q}_t) = \partial f / \partial \boldsymbol{q} | \boldsymbol{q} = \boldsymbol{q}_t$ its Jacobian. We can now obtain a Gaussian task likelihood by approximating the forward kinematics by its linearization through the Jacobian, i.e., $\boldsymbol{x} \approx f(\boldsymbol{q}_0) + J(\boldsymbol{q} - \boldsymbol{q}_0)$. The parameters of the Gaussian are then given by $\boldsymbol{r}_t = J^T C\big(f(\boldsymbol{q}_0) - \boldsymbol{x}^*\big)$ and $R_t = J^T C J$, where the diagonal elements of the matrix C specify the desired precision in task space.

Algorithm 1: Approximate Inference Control with Regularized Update Steps

Input: initial state $\boldsymbol{q}_0$, parameter $\alpha^{[0]}$, threshold θ
Output: feedback control law $\boldsymbol{o}_{0:T-1}$ and $O_{0:T-1}$
initialize $\boldsymbol{q}_{1:T}^{[0]} = \boldsymbol{q}_0$, $S_0 = 1e10 \cdot \mathbf{I}$, $\boldsymbol{s}_0 = S_0 \boldsymbol{q}_0$, $n = 1$
while *not converged* **do**
  $\boldsymbol{q}_{0:T}^{[n-1]} = \boldsymbol{q}_{0:T}^{[n]}$
  for $t \leftarrow 1$ **to** T **do**
    linearize model: $A_t, \boldsymbol{a}_t, B_t$
    compute: $H_t, \boldsymbol{h}_t, R_t, \boldsymbol{r}_t$
    update: $\boldsymbol{s}_t$, S_t, $\boldsymbol{v}_t$, V_t, $\boldsymbol{b}_t$, and B_t
    if $\|\boldsymbol{b}_t - \boldsymbol{q}_t^{[n]}\| > \theta$ **then**
      repeat this time step
      $t \leftarrow t - 1$
    $\boldsymbol{q}_t^{[n]} = B_t^{-1} \boldsymbol{b}_t$
  for $t \leftarrow T - 1$ **to** 0 **do**
    ..same updates as above...
  for $t \leftarrow 0$ **to** $T - 1$ **do**
    compute feedback controller: $\boldsymbol{o}_t, O_t$
    $\boldsymbol{u}_t^{[n]} = \boldsymbol{o}_t + O_t \boldsymbol{q}_t$
    $\boldsymbol{q}_{t+1}^{[n]} = A_t \boldsymbol{q}_t^{[n]} + \boldsymbol{a}_t + B_t \boldsymbol{u}_t^{[n]}$
  $n = n + 1$
  $\alpha^{[n]} = \alpha^{[n-1]} \gamma$
return $\boldsymbol{o}_{0:T-1}$ and $O_{0:T-1}$

C. Evaluation of the convergence properties of AICO

To investigate the convergence properties of AICO, we use a simulated light-weight robot arm [22] with five joints. The robot has to reach a desired end effector position in Cartesian space and subsequently has to return to its initial pose. To increase the complexity, we define a second task, where the robot should additionally keep the roll angle of the end effector constant. For this task, we used the cost function

$$C_t(\boldsymbol{q}_t, \boldsymbol{u}_t) = \begin{cases} 10^4 (\boldsymbol{x}^i - \boldsymbol{x}_t)^T (\boldsymbol{x}^i - \boldsymbol{x}_t) & \text{if } t = T^i \\ 10^{-2} \boldsymbol{u}^T \boldsymbol{u} & \text{else} \end{cases} , \quad (6)$$

where $\boldsymbol{x}^i$ denotes the desired robot postures in task space at times $T^1 = 500$ and $T^2 = 10^3$ (the planning horizon is 2 seconds with a time step of 2ms) with $\boldsymbol{x}^1 = [1, -0.4, 0, 0, \pi/2, 0]^T$ and $\boldsymbol{x}^2 = [1, 0, 0, 0, \pi/2, 0]^T$. Note that we do not assume any initial solution to initialize the planner, solely the initial posture of the robot in configuration space is used as initial 'trajectory'. An example movement is shown in Figure 1.

Using the *pseudo-dynamics* approximation of the system dynamics, the convergence rate of the costs per iteration of both tasks are shown in Figure 2A,B. For the simple task in Figure 2A the inferred cost values converge fast for all algorithms, with the standard AICO algorithm showing

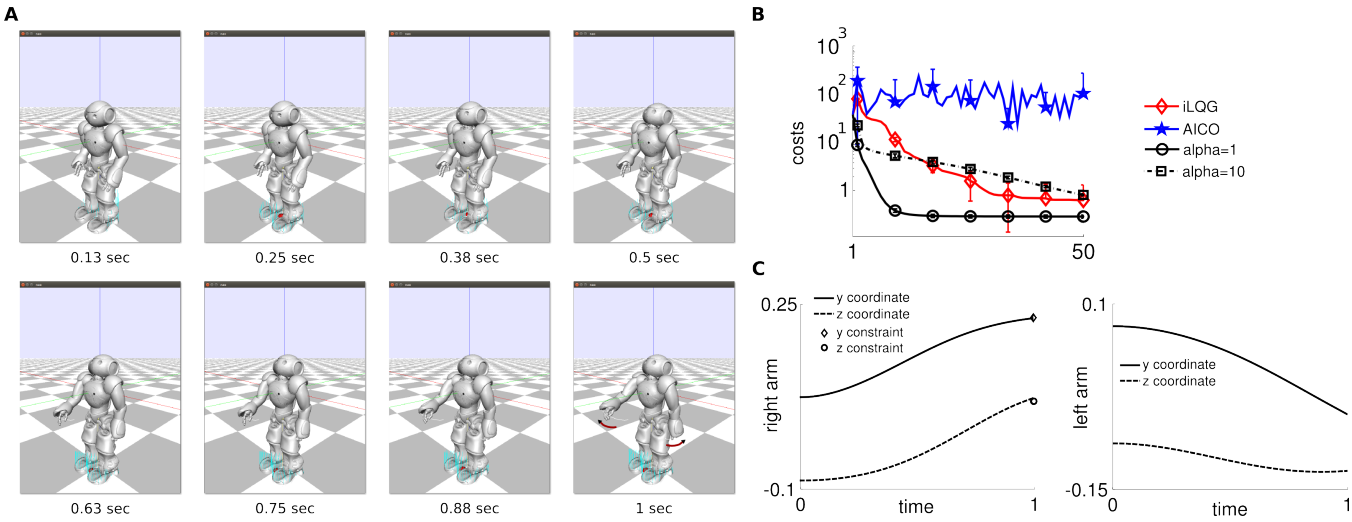

Fig. 3. Reaching task with the humanoid robot *Nao*. The robot has to reach a desired end effector position with the right arm while maintaining balance. Eight snapshots of the inferred movement are shown in (A). In (B), the convergence of the costs of the optimization procedure is shown, where we compare *iLQG*, the standard implementation of AICO and the regularized variant. The mean and the standard deviations for 10 initial states '$\boldsymbol{q}_0$ are sampled from a Gaussian with zero mean and a standard deviation of 0.05. The movement objectives for the right arm are shown in the left panel in (C). To counter balance, the robot moves its left hand and the head.

the best performance. However, the fast convergence also comes with the costs of a reduced robustness of the policy update as can be seen from the results in the second scenario illustrated in Figure 2B, where AICO is unstable and cannot infer solutions with low costs. When we used the analytic forward dynamic model (where the linearizations are computed through finite differences) instead of the pseudo dynamics model, computing the messages in AICO became numerically unstable and no solutions could be inferred. Therefore, the panels in Figure 2C,D do not include results of AICO. We also evaluated the *iLQG* method [7] that implements an adaptive regularization schedule and line search to prevent divergence [8]. While the *iLQG* algorithm performed well for the pseudo dynamics model, the learning rate was automatically decreased to almost zero for the analytical dynamics model. Our regularization method for AICO, that we will present in the next section, considerably outperformed both competing methods.

D. Regulating the policy updates in AICO

To regularize the policy update steps in (1), we add an additional cost term to the task-likelihood function, i.e.,

$$p(r_t = 1|\boldsymbol{q}_t^{[n]}, \boldsymbol{u}_t^{[n]}) \propto \exp\{-C_t(\boldsymbol{q}_t^{[n]}, \boldsymbol{u}_t^{[n]}) - \alpha^{[n]}(\boldsymbol{q}_t^{[n]} - \boldsymbol{q}_t^{[n-1]})^T(\boldsymbol{q}_t^{[n]} - \boldsymbol{q}_t^{[n-1]})\} \quad ,$$

which punishes the distance of the state trajectories of two successive iterations of the algorithm (n denotes the iteration). The parameter α controls the size of the update step. For large α, the trajectory update will be conservative as the algorithm will stay close to the previous trajectory that has been used for linearization. For small α values, the new trajectory will directly jump to the LQG solution given the linearized dynamics and the approximated costs. Hence, α is inverse proportional to the step size. The value of α is updated after each iteration according to $\alpha^{[n]} = \alpha^{[n-1]}\gamma$. For $\alpha^{[0]} \geq 1$ and $\gamma > 1$, convergence is guaranteed as the regularization term will dominate with an increasing number of iterations.

The algorithms is listed in Algorithm 1. An interesting feature of this algorithm is that no learning rate is needed as α is used directly to implement a step size. In the original formulation of AICO the learning rate is either applied to the state update (in Line 13 in Algorithm 1) [9] or to the feedback controller (in Line 18 in Algorithm 1) [10]. However, neither implementation can guarantee convergence in nonlinear systems or in tasks with costs inducing a nonlinear mapping from $\mathbb{Q}$ to $\mathbb{X}$.

We evaluate the resulting algorithm on the same robot arm reaching tasks. For both tasks, the Cartesian planning task in Figure 2A,C and the extension with the additional roll angle objective in Figure 2B,D, we evaluated AICO with the regularization parameter $\alpha \in \{1, 10\}$ (we did not increase α and $\gamma = 1$). For both models of the system dynamics, the *pseudo-dynamics* approximation (shown in Figure 2A,B) and the analytic model (illustrated in Figure 2C,D), AICO benefits from the regularization term and the costs decay exponentially fast. Interestingly, without "good" initial solutions, the differential dynamic programming method *iLQG* [8] that implements a sophisticated regularization scheme cannot generate movement policies with low costs when using the analytic model. This is shown in Figure 2C,D.

III. RESULTS

We evaluated the proposed planning method in simulation with the humanoid robot Nao. The Nao robot has 25 degrees-of-freedom. In first experiments, we investigated the performance of the planner with a *pseudo-dynamics* model

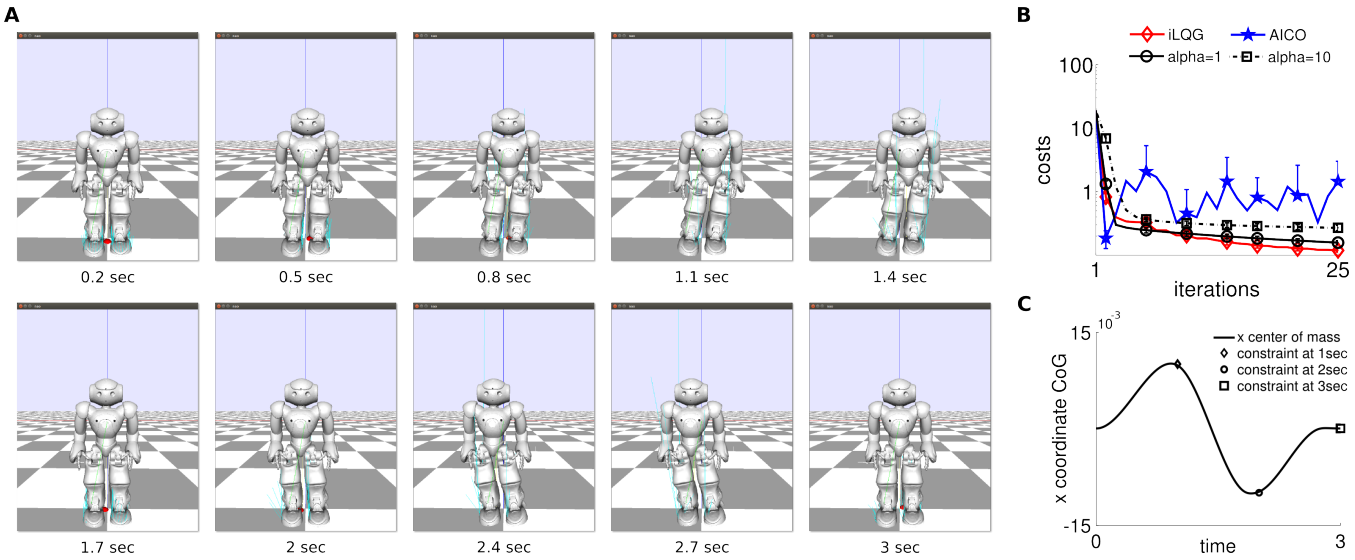

Fig. 4. Balancing task in the humanoid robot *Nao*. The robot should *swing* its hips, which is encoded by adding an offset scalar to the x-coordinate of the center of gravity vector. In (A) 10 snapshots of the resulting movement for an increasing planning horizon are shown for $\alpha = 1$. The convergence properties of *iLQG*, the standard AICO and its regularized variants are shown in (B). The mean and the standard deviations for 10 initial states '$\boldsymbol{q}_0$ are sampled from a Gaussian with zero mean and a standard deviation of 0.05. In (C) the x-coordinate of the center of gravity of the Nao is illustrated. The large dots denote the objectives.

of the robot.

The humanoid had to reach for an end effector target using the right arm while maintaining balance. In a second experiment, Nao had to shift the x-coordinate of the center of gravity while maintaining balance.

A. Arm reaching with a humanoid robot

The humanoid has to reach for the end effector target $\boldsymbol{x}^* = [0, 0.2, 0.06]^T$, where only the y- and the z- Cartesian coordinates are relevant. Additionally, the robot has to maintain balance, which is implemented as deviation of the center of gravity vectors from its initial values $\boldsymbol{x}_{\text{CoG}}(t=0)$, i.e., we specify the desired center of gravity as $\boldsymbol{x}^*_{\text{CoG}} = \boldsymbol{x}_{\text{CoG}}(t=0)$. The same cost function as in the experiments for the light weight robot arm in (6) is used. For this task, however, only a single via-point is defined that is used for the desired end effector target and the center of gravity, i.e., $\boldsymbol{x}^1 = [\boldsymbol{x}^{*T}, \boldsymbol{x}^*_{\text{CoG}}{}^T]^T$.

Only by specifying two scalars in $\boldsymbol{x}^*$ (the scaling parameters in (6) are constants that take the values 10^4 or 10^{-2}), the planning algorithm infers 50-dimensional state trajectories (the state $\boldsymbol{q}_t$ at time t encodes the joint angles and the joint velocities, ignoring the base frame). This is shown in Figure 3A for the proposed planning algorithm with the regularization parameter $\alpha = 1$. As in the robot arm experiments, the Approximate Inference Control (AICO) algorithm benefits from the regularization. As can be seen in Figure 3B, AICO cannot infer movement solutions with low costs without regularization.

Interestingly, to maintain balance, the humanoid utilizes its head and its left arm for which no objectives were explicitly specified. This effect is a feature of model-based planning methods that consider the coupled dynamics and is best illustrated in Figure 3C, where the end effector trajectories of both arms and the desired target values are shown.

B. Balancing with a humanoid

In this task the humanoid has to balance on one foot by moving its center of gravity. In this experiment, we specify three desired via-points for the center of gravity, i.e., $\boldsymbol{x}^i = \boldsymbol{x}^i_{\text{CoG}}$ with $i = 1, ..., 3$. The last via-point is set to the initial center of gravity $\boldsymbol{x}_{\text{CoG}}(t=0)$. The first via-point has an offset of 0.1m in the x-coordinate of $\boldsymbol{x}_{\text{CoG}}(t=0)$ to force the robot to move its center of gravity to the right. The second via point has the same negative offset in the x-direction to exhibit a movement to the left. The planning horizon was three seconds ($T^1 = 100$,$T^2 = 200$ and $T^3 = 300$ with $\tau = 10$ms) and the distance matrix C in (6) was scaled with the importance weights $[10^6, 10, 10]^T$ for the x,y, and z coordinate of $\boldsymbol{x}^i_{\text{CoG}}$.

For $\alpha = 1$, the resulting movement is illustrated in Figure 4A. Illustrated are 10 snapshots. Nao first moves its hip to the right (with respect to the robot frame) and thereafter to the left. This movement is the result of an inference problem encoded in mainly two scalars, i.e., the offsets.

The standard implementation of AICO was not able to infer successful balancing solutions, which is illustrated in Figure 4B. In contrast, the regularized variant using $\alpha \in \{1, 10\}$ converged after 25 iterations of the trajectory optimization procedure. For $\alpha = 1$, the x-coordinate of the center of gravity and the implemented objectives are shown in 4C.

C. Computational time

The computational time of the proposed planning algorithm is the same as for the standard implementation of

AICO. If the algorithm is implemented in C-code it achieves real time performance in humanoid planning problems [9]. However, for our experiments we used a Matlab implementation on a standard computer (2.4GHz, 8GB RAM), where, e.g., the computation of the balancing movements in Figure 4 took less then 50 seconds (which includes all 25 iterations of the optimization process). The movement duration of the executed trajectory was three seconds.

IV. CONCLUSIONS

Stochastic Optimal Control (SOC) methods are powerful planning methods to infer high-dimensional state and control sequences [7]–[9], [20]. For real time applications in humanoids, efficient model predictive control variants have been proposed [8]. However, the quality of the generated solutions heavily depends on the initial movement policy and on the accuracy of the approximations of the system dynamics. Most methods use regularization to prevent numerical instabilities, but typically greedily exploit the approximated system dynamics model. The resulting trajectory update might be far from the previous trajectory used for linearization.

As the linearizations are only locally valid, we explicitly avoid large jumps in the trajectories by punishing large deviations from the previous trajectory. We demonstrated in this paper that SOC methods can greatly benefit from such a regularization term. We used such regularization term for the Approximate Inference Control (AICO) algorithm [9]. Due to the regularization term, which implicitly specifies the step size of the trajectory update, no learning rate as in the standard formulation of AICO is needed. Our experiment shows that the used regularization term considerably outperforms existing SOC methods that are based on linearization, in particular if highly non-linear system dynamics are used.

An interesting open question is if the proposed regularization parameter facilitates a combination of SOC and model learning approaches. Typically, inaccurate model predictions have catastrophic effects on the numerical stability of SOC methods. In particular, if the model predictions are poor, the SOC method should not further explore but collect more data around the current trajectory. Such idea could be implemented by modeling the regularization parameter as a function of the model uncertainty.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements No. 270327 (CompLACS) and No. 600716 (CoDyCo). The authors would like to acknowledge Guilherme Maeda, Tucker Hermans and Serena Ivaldi for their assistance during the preparation of this manuscript.

REFERENCES

[1] Paolo Baerlocher and Ronan Boulic. Task-priority formulations for the kinematic control of highly redundant articulated structures. In *IROS*, pages 13–17, 1998.

[2] Su Il Choi and Byung Kook Kim. Obstacle avoidance control for redundant manipulators using collidability measure. *Robotica*, pages 143–151, 2000.

[3] Tomomichi Sugihara and Yoshihiko Nakamura. Whole-body cooperative balancing of humanoid robot using cog jacobian. In *IROS*, pages 2575–2580, 2002.

[4] Michael Gienger, Herbert Janssen, and Christian Goerick. Task-oriented whole body motion for humanoid robots. In *IEEE-RAS*, pages 238–244, 2005.

[5] Koichi Nishiwaki, Mamoru Kuga, Satoshi Kagami, Masayuki Inaba, and Hirochika Inoue. Whole-body cooperative balanced motion generation for reaching. *IJHR*, pages 437–457, 2005.

[6] Robert F Stengel. *Stochastic optimal control: theory and application*. John Wiley & Sons, Inc., 1986.

[7] Emanuel Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *ACC*, pages 300–306, 2005.

[8] Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *IROS*, pages 4906–4913, 2012.

[9] Marc Toussaint. Robot trajectory optimization using approximate inference. In *ICML*, pages 1049–1056, 2009.

[10] Elmar Rueckert and Gerhard Neumann. Stochastic optimal control methods for investigating the power of morphological computation. *Artificial Life*, pages 115–131, 2013.

[11] Marc Toussaint and Amos Storkey. Probabilistic inference for solving discrete and continuous state markov decision processes. In *ICML*, pages 945–952, 2006.

[12] Matthew D Hoffman, Nando D Freitas, Arnaud Doucet, and Jan R Peters. An expectation maximization algorithm for continuous markov decision processes with arbitrary reward. In *AISTATS*, pages 232–239, 2009.

[13] Marc Toussaint, Nils Plath, Tobias Lang, and Nikolay Jetchev. Integrated motor control, planning, grasping and high-level reasoning in a blocks world using probabilistic inference. In *ICRA*, pages 385–391, 2010.

[14] Marc Toussaint and Christian Goerick. Probabilistic inference for structured planning in robotics. In *Int. Conf. on Intelligent Robots and Systems (IROS 2007)*, pages 3068–3073, 2007.

[15] Thomas Furmston and David Barber. Variational methods for reinforcement learning. In *AISTATS*, pages 241–248, 2010.

[16] Marc Toussaint and Christian Goerick. A bayesian view on motor control and planning. In *From Motor Learning to Interaction Learning in Robots*, pages 227–252. Springer, 2010.

[17] Hilbert J Kappen, Vicenç Gómez, and Manfred Opper. Optimal control as a graphical model inference problem. *JMLR*, pages 159–182, 2012.

[18] J. Peters and S. Schaal. Learning operational space control. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[19] Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *ICML*, pages 745–750, 2007.

[20] Konrad Rawlik, Marc Toussaint, and Sethu Vijayakumar. An approximate inference approach to temporal optimization in optimal control. In *NIPS*, pages 2011–2019, 2010.

[21] Jur Van Den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *IJRR*, pages 1263–1278, 2012.

[22] Thomas Lens, Jürgen Kunz, Oskar von Stryk, Christian Trommer, and Andreas Karguth. Biorob-arm: A quickly deployable and intrinsically safe, light-weight robot arm for service robotics applications. In *ISR/ROBOTIK*, pages 1–6, 2010.