Robot Trajectory Optimization using Approximate Inference

Marc Toussaint MTOUSSAI@CS.TU-BERLIN.DE
TU Berlin, Franklinstr 28/29 FR6-9, 10587 Berlin, Germany

Abstract

The general stochastic optimal control (SOC) problem in robotics scenarios is often too complex to be solved exactly and in near real time. A classical approximate solution is to first compute an optimal (deterministic) trajectory and then solve a local linear-quadratic-gaussian (LQG) perturbation model to handle the system stochasticity. We present a new algorithm for this approach which improves upon previous algorithms like iLQG. We consider a probabilistic model for which the maximum likelihood (ML) trajectory coincides with the optimal trajectory and which, in the LQG case, reproduces the classical SOC solution. The algorithm then utilizes approximate inference methods (similar to expectation propagation) that efficiently generalize to non-LQG systems. We demonstrate the algorithm on a simulated 39-DoF humanoid robot.

1. Introduction

Trajectory optimization is a key problem in robotics, in particular when the time needed for optimization is critical. For instance, in interactive scenarios with a human, robots need to react on dynamic changes of the environment and the optimization time should be no more than the temporal duration of the movement itself so that an online sequencing is in principle possible. The aim of this work is to push the performance of trajectory optimization algorithms a little further using approximate inference techniques.

The transfer of inference methods to stochastic control and Reinforcement Learning (RL) problems has a long history. Examples in the context of decision making and RL include the early work on inference in influence diagrams (Cooper, 1988; Shachter, 1988), the relation between expectation-maximization and (immediate reward) RL observed by (Dayan & Hinton, 1997), Attias' (2003) inference approach to planning, and an EM approach to solving MDPs and POMDPs (Toussaint et al., 2006). In the control literature, the relation between stochastic optimal control (SOC) and inference (Kalman estimation) is long known for the special LQG case (e.g., Stengel, 1986). However, one of the most interesting and not fully solved questions in the field is how this transfer can be generalized to non-LQG problems – we will discuss such work in some more detail below.

In this paper we do not try to propose an exact and general reformulation of SOC problems as an inference problem. Instead, in its present form (focusing on Gaussian messages), our approach aims at a local approximate solution to the SOC problem that is fast to compute. This is in the tradition of the classical SOC literature (see, e.g, Bryson & Ho, 1969; Stengel, 1986), where the LQG case is typically introduced as a (variational) model to control perturbations from an optimal trajectory. That is, the general non-linear SOC problem is approximately solved by first computing an optimal trajectory for the noise-free system, and then deriving a local LQG model of the perturbations around this optimal trajectory. As long as the system is not perturbed too far from the optimal trajectory, the corresponding linear quadratic regulator (LQR) is a reasonable solution to the SOC problem. The 2nd stage of computing the local LQR is analytically solved by the Ricatti equations (see below); the main problem arises with the non-linear trajectory optimization problem in the first stage.

The trajectory optimization problem can be solved, e.g., by gradient methods (typically with a spline-based trajectory encoding, e.g., (Chen, 1991; Zhang & Knoll, 1995; Schlemmer & Gruebel, 1998)) or by sequential quadratic programming (SQP) schemes. In every iteration of SQP one devices a quadratic approximation of the objective (cost) function – the optimum of which can be computed using the Ricatti equations. Hence, SQP typically involves iterating LQG solution methods. An instance of such an algorithm is iLQG

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009. Copyright 2009 by the author(s)/owner(s).

(Todorov & Li, 2005). An aspect of such methods which will contrast to our approach is that in each SQP iteration one computes a global trajectory $x_{0:T}$ (by "global" we mean "over the full time interval $0..T$") and a global LQG approximation. Our framework will naturally lead to iterative updates of local messages rather than a global trajectory.

In the LQG case, the similarity between the Ricatti equations and Kalman's equation for state estimation is referred to as "Kalman duality" (Stengel, 1986; Todorov, 2008) – but it strictly holds only in the LQG case. The crucial question is how to generalize this duality to the non-LQG case. Todorov (2008) introduces a special case of SOC problems which can be solved via inference (see also (Kappen et al., 2009)). An alternative to seeking an exact duality is to focus on optimal trajectories and consider a probabilistic model for which the ML solution coincides with the optimal trajectory (Todorov, 2008). We follow this latter approach and will utilize approximate inference methods to efficiently find the ML trajectory and the local LQG solution around this trajectory. In previous work (Toussaint & Goerick, 2007) we applied the same idea for a special case of coupled task and joint space planning. In this paper we focus on the general theory and SOC case.

In the next section we will briefly introduce the framework of SOC, the LQG case and the SQP method iLQG. In section 3 we formulate the general probabilistic model. We derive an exact Gaussian message passing algorithm for the special LQG case and show how it is related to the Ricatti equations. Then we generalize to the non-LQG case based on Expectation Propagation (Minka, 2001b). Finally, section 4 presents an evaluation on a simulated humanoid reaching problem.

2. Stochastic optimal control

We consider a discrete time stochastic controlled system of the form

$$x_{t+1} = f_t(x_t, u_t) + \xi \ , \quad \xi \sim \mathcal{N}(0, Q_t) \ , \tag{1}$$

with time step t, state $x_t \in \mathbb{R}^n$, control $u_t \in \mathbb{R}^m$, and Gaussian noise ξ of covariance Q. We also use the notation $P(x_{t+1} \,|\, u_t, x_t) = \mathcal{N}(x_{t+1} \,|\, f_t(x_t, u_t), Q_t)$, where

$$\mathcal{N}(x|a, A) \propto \exp\{-\frac{1}{2}(x-a)^\top \ A^{-1} \ (x-a)\} \tag{2}$$

is a Gaussian over x with mean a and covariance A. For a given state-control sequence $x_{0:T}, u_{0:T}$ we define the cost as

$$C(x_{0:T}, u_{0:T}) = \sum_{t=0}^{T} c_t(x_t, u_t) \ . \tag{3}$$

The optimal value function $J_t(x)$ gives the expected future cost when in state x at time t for the best controls and obeys the Bellman optimality equation

$$J_t(x) = \min_u \Big[c_t(x, u) + \int_{x'} P(x' \,|\, u, x) \ J_{t+1}(y)\Big] \ . \tag{4}$$

There are two versions of stochastic optimal control problems: The open-loop control problem is to find a control sequence $u^*_{1:T}$ that minimizes the expected cost. The closed-loop (feedback) control problem is to find a control policy $\pi^*_t : x_t \mapsto u_t$ (that exploits the true state observation in each time step and maps it to a feedback control signal) that minimizes the expected cost.

The linear quadratic gaussian (LQG) case plays an important role as a perturbation model or as an ingredient in iterative solution methods. LQG is a linear control process with Gaussian noise,

$$P(x_{t+1} \,|\, x_t, u_t) = \mathcal{N}(x_{t+1} \,|\, A_t x_t + a_t + B_t u_t, Q_t) \ ,$$

and quadratic costs,

$$c_t(x_t, u_t) = x_t^\top R_t x_t - 2r_t^\top x_t + u_t^\top H_t u_t \ . \tag{5}$$

The LQG process is defined by matrices and vectors $A_{0:T}, a_{0:T}, B_{0:T}, Q_{0:T}, R_{0:T}, r_{0:T}, H_{0:T}$. As a convention, in the remainder of this paper we will drop the subscript $_t$ for A, a, B, Q, R, r, H – wherever a subscript is missing we refer to time t.

The LQG case allows us to derive an exact backward recursion, called Ricatti equation, for the computation of the value function. The value function will always be a quadratic form of the state. Assume that at time $t{+}1$ the optimal value function can be expressed as

$$J_{t+1}(x) = x^\top V_{t+1} x - 2v_{t+1}^\top x \tag{6}$$

for some matrix V_{t+1} and some vector v_{t+1}. Applying equation (4) in a straight-forward manner one can derive that $J_t(x)$ is of the form

$$\begin{aligned} J_t(x) &= x^\top V_t x - 2x^\top v_t + \textit{terms independent of } x \ , \\ V_t &= R + A^\top V_{t+1} A - K V_{t+1} A & (7) \\ v_t &= r + A^\top(v_{t+1} - V_{t+1} a) - K(v_{t+1} - V_{t+1} a) & (8) \\ K &:= A^\top V_{t+1}^\top (V_{t+1} + B^{-\top} H B^{-1})^{-1} \ , \end{aligned}$$

and the minimization in (4) is given by

$$u^*_t(x) = -(H + B^\top V_{t+1} B)^{-1} B^\top (V_{t+1}(Ax + a) - v_{t+1}) \ . \tag{9}$$

Equations (7)-(9) are the Ricatti equations (Stengel, 1986). (Using the Woodbury identity it can be rewritten in other forms.) Initialized with $V_T = R_T$ and

Algorithm 1 iLQG

```
1: Input: initial trajectory $x_{0:T}$, convergence rate $\alpha$,
   control costs $H_{0:T}$, functions $A_t(x)$, $a_t(x)$, $B_t(x)$,
   $R_t(x)$, $r_t(x)$
2: Output: trajectory $x_{0:T}$, LQR $V_{0:T}, v_{0:T}$
3: repeat
4:   access $R_T(x_T)$, $r_T(x_T)$
5:   $V_T \leftarrow R_T$, $v_T \leftarrow r_T$
6:   for $t = T-1$ to $0$ do //bwd Ricatti recursion
7:     access $A_t(x_t)$, $a_t(x_t)$, $B_t(x_t)$, $R_t(x_t)$, $r_t(x_t)$
8:     compute $V_t$ and $v_t$ using (7,8)
9:   end for
10:  for $t = 0$ to $T-1$ do //fwd control recursion
11:    compute $u^*_t(x_t)$ using (9)
12:    $x_{t+1} \leftarrow (1-\alpha)x_{t+1} + \alpha[A_t x_t + a_t + B_t u^*(x_t)]$
13:  end for
14: until convergence
```

$v_T = r_T$ this gives a backward recursion to compute the value function J_t at each time step. The linear quadratic regulator (LQR) in (9) gives the optimal control policy, i.e., a solution to the closed-loop problem in the LQG case. Note that the optimal control and path is independent of the process noise Q.

As mentioned in the introduction, the LQG case is classically introduced as a perturbation model around an optimal trajectory of the deterministic system (Bryson & Ho, 1969; Stengel, 1986). Alternatively, the LQG case can also be applied as a means to find this optimal trajectory: Optimizing the trajectory $x_{0:T}$ can be addressed in the manner of sequential quadratic programming, where one starts with a guess of $x_{0:T}$, then computes a 2nd order approximation of the cost around this initialization, uses an exact solver (LQG in this case) to jump to the optimum of this 2nd order approximation, and iterates. This method, termed iLQG by (Todorov & Li, 2005), computes a (locally) optimal trajectory and as an aside also provides the corresponding LQR around this trajectory which can be used to handle perturbations. Algorithm 1 makes this procedure explicit and includes an additional convergence rate parameter which increases robustness greatly.

3. Probabilistic inference approach

The classical approach to design a good trajectory is to define a cost function (e.g., penalizing collisions, rewarding goals) and minimize the expected cost given a stochastic control model. An alternative take on defining what a good trajectory is to condition a probabilistic trajectory model on desired criteria (e.g., conditioning on not observing collision, conditioning on reaching a goal) and consider the problem of inferring the posterior distribution of trajectories conditioned on these criteria. In general these two ways of defining optimality are not fully equivalent (there is yet no general version of Kalman's duality known) – but both are interesting definitions of optimality. Accordingly, the formulation we present below will not in all respects be equivalent to the problem of expected cost minimization, but we can ensure that the ML trajectory of the conditioned probabilistic model coincides with the optimal trajectory in the classical case.

Let us introduce an additional binary random variable z_t in the process with conditional probability

$$P(z_t\!=\!1 \mid u_t, x_t) = \exp\{-c_t(x_t, u_t)\} \ . \qquad (10)$$

As mentioned in the introduction, this idea has a long history (Cooper, 1988; Shachter, 1988; Dayan & Hinton, 1997). Shachter and Peot (1992) even mention work by Raiffa (1969) and von Neumann & Morgenstern (1947) in this context. In our definition (10), the costs $c_t(x_t, u_t)$ play the role of the neg-log-probability of $z_t = 1$, in accordance to the typical identification of energy with neg-log-probability – which differs from the above cited approaches in which a binary random variable with probability proportional to reward or utility is defined. The neg-log approach will lead us directly to an inference version of LQG for which we can exploit approximate inference techniques.

Clearly, for a *single* state-control trajectory $x_{0:T}, u_{0:T}$, the log-likelihood

$$\log P(z_{0:T}\!=\!1 \mid x_{0:T}, u_{0:T}) = -C(x_{0:T}, u_{0:T}) \qquad (11)$$

is the negative classical cost (3) – hence an ML trajectory coincides with the classical optimal trajectory. However, taking the expectation over trajectories,

$$\begin{aligned}
&\log P(z_{0:T}\!=\!1)\\
&\quad= \log \mathrm{E}_{u_{0:T},x_{0:T}}\{\prod_{t=0}^{T} P(z_t\!=\!1 \mid u_t, x_t)\}\\
&\quad\ge \mathrm{E}_{u_{0:T},x_{0:T}}\{\log \prod_{t=0}^{T} P(z_t\!=\!1 \mid u_t, x_t)\} &(12)\\
&\quad= \mathrm{E}_{u_{0:T},x_{0:T}}\{-\sum_{t=0}^{T} c_t(u_t, x_t)\}\\
&\quad= -\mathrm{E}_{u_{0:T},x_{0:T}}\{C(u_{0:T}, x_{0:T})\} \ , &(13)
\end{aligned}$$

we find that likelihood maximization is in general not equivalent to expected cost minimization. In the LQG case there exists a concise relation, as we show in the next section.

3.1. LQG case and equivalence to Ricatti

In the LQG case the costs $c_t(x_t, u_t)$ are quadratic in x_t and u_t as given by (5). In terms of the binary random variable z_t this translates to Gaussian probabilities

$$P(z_t\!=\!1\,|\,x_t) \propto \mathcal{N}[x_t\,|\,r_t, R_t]\ , \tag{14}$$

$$P(u_t) = \mathcal{N}[u_t\,|\,0, H]\ , \tag{15}$$

where we interpreted the quadratic cost over u_t as a prior and let z_t only depend on x_t (which is equivalent but more convenient than a uniform prior over u_t and $P(z_t\!=\!1\,|\,u_t, x_t) \propto \mathcal{N}[x_t\,|\,r_t, R_t]\mathcal{N}[u_t\,|\,0, H]$). The bracket notation

$$\mathcal{N}[x|a, A] \propto \exp\{-\frac{1}{2}x^\top\ A\ x + x^\top a\} \tag{16}$$

denotes a Gaussian over x in canonical form, with precision matrix A and mean $A^{-1}a$.

We can simplify by integrating out the control u_t,

$$\begin{aligned} &P(x_{t+1}\,|\,x_t) \\ &\quad = \textstyle\int_u du\ \mathcal{N}(x_{t+1}\,|\,Ax_t + a + Bu_t, Q)\ \mathcal{N}[u_t\,|\,0, H] \\ &\quad = \mathcal{N}(x_{t+1}\,|\,Ax_t + a, Q + BH^{\text{-}1}B^\top)\ . \end{aligned} \tag{17}$$

Consider the problem of computing the posterior distribution $P(x_{0:T}\,|\,z_{0:T}\!=\!1)$ over state trajectories conditioned on that we permanently observe $z_t = 1$. Inference is a standard forward-backward process. We prefer to derive this inference process in terms of a message passing algorithm[1] because this will allow us most directly to generalize the algorithm to the non-linear case.

Theorem 1. *The messages in the LQG process defined by equations (17) and (14) are*

$$\mu_{x_{t\text{-}1}\to x_t}(x_t) = \mathcal{N}(x_t\,|\,s_t, S_t) \tag{20}$$

$$s_t = a_{t\text{-}1} + A_{t\text{-}1}(S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1})^{\text{-}1}(S_{t\text{-}1}^{\text{-}1}s_{t\text{-}1} + r_{t\text{-}1})$$

$$S_t = Q + B_{t\text{-}1}H^{\text{-}1}B_{t\text{-}1}^\top + A_{t\text{-}1}(S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1})^{\text{-}1}A_{t\text{-}1}^\top$$

$$\mu_{x_{t+1}\to x_t}(x_t) = \mathcal{N}(x_t\,|\,v_t, V_t) \tag{21}$$

$$v_t = -A_t^{\text{-}1}a_t + A_t^{\text{-}1}(V_{t+1}^{\text{-}1} + R_{t+1})^{\text{-}1}(V_{t+1}^{\text{-}1}v_{t+1} + r_{t+1})$$

$$V_t = A_t^{\text{-}1}[Q + B_tH^{\text{-}1}B_t^\top + (V_{t+1}^{\text{-}1} + R_{t+1})^{\text{-}1}]A_t^{\text{-}\top}$$

$$\mu_{z_t\to x_t}(x_t) = \mathcal{N}[x_t\,|\,r_t, R_t] \tag{22}$$

The straight-forward derivation of these messages is given in the appendix.

The backward messages are closely related to the Ricatti equation as we will show below. However, note that there is no counterpart of the forward messages in the classical approaches. This is crucial. The forward messages are necessary to estimate a proper posterior belief of a state and this belief will play an important role in approximate methods below. Concerning the backward messages, let us define

$$\bar V_{t+1} = V_{t+1}^{\text{-}1} + R_{t+1} \tag{23}$$

$$\bar v_{t+1} = V_{t+1}^{\text{-}1}v_{t+1} + r_{t+1}\ , \tag{24}$$

which corresponds to a backward message (in canonical representation) which has the cost message already absorbed. Using a special case of the Woodbury identity,

$$(A^{\text{-}1} + B)^{\text{-}1} = A - A(A + B^{\text{-}1})^{\text{-}1}A\ , \tag{25}$$

the bwd messages can be rewritten as

$$\begin{aligned} V_{t+1}^{\text{-}1} &= A^\top[\bar V_{t+1}^{\text{-}1} + Q + BH^{\text{-}1}B^\top]^{\text{-}1}A \\ &= A^\top\bar V_{t+1}A - K\bar V_{t+1}A \end{aligned} \tag{26}$$

$$K := A^\top\bar V_{t+1}[\bar V_{t+1} + (Q + BH^{\text{-}1}B^\top)^{\text{-}1}]^{\text{-}1}$$

$$\begin{aligned} V_t^{\text{-}1}v_t &= -A^\top\bar V_{t+1}a_t + A^\top\bar v_{t+1} + K\bar V_{t+1}a_t - K\bar v_{t+1} \\ &= A^\top(\bar v_{t+1} - \bar V_{t+1}a_t) - K(\bar v_{t+1} - \bar V_{t+1}a_t) \end{aligned} \tag{27}$$

$$\bar V_t = R_t + (A^\top - K)\bar V_{t+1}A \tag{28}$$

$$\bar v_t = r_t + (A^\top - K)(\bar v_{t+1} - \bar V_{t+1}a_t) \tag{29}$$

They correspond exactly to the Recatti equations (7), (8) except for the dependence on Q which interacts directly with the control cost metric H.

3.2. Approximate inference in the non-LQG case

Let X be a random variable for which we can express the belief

$$b(X) = \prod_i t_i(X) \tag{30}$$

as a product of incoming messages $t_i(X)$. Expectation Propagation (Minka, 2001b) is a method to iteratively improve the messages (and thereby the belief) when they are bound to be approximate. Let $\mathcal{F}$ be a family of distributions. We constrain every approximate message $\hat t_i \in \mathcal{F}$ to be in this family of distributions, and

[1] We briefly recap the definition of messages in pair-wise factor graphs; see (Yedidia et al., 2001; Murphy, 2002; Minka, 2001a) for a thorough introduction. Given two random variables X_i and X_j coupled by a pair-potential $f_{ij}(X_i, X_j)$, the message passing equations are

$$\mu_{j\to i}(X_i) = \sum_{X_j} f_C(X_i, X_j) \prod_{k:k\neq i} \mu_{k\to j}(X_j)\ , \tag{18}$$

where k indicates variables coupled to j other than i. Given all incoming messages to a variable, the posterior marginal belief is given as their product,

$$b_i(X_i) := \prod_j \mu_{j\to i}(X_i)\ . \tag{19}$$

hence also the approximate belief $\hat{b} \in \mathcal{F}$. We update a message $\hat{t}_i$ in such a way that the updated belief $\hat{b}'$ is as close as possible to "when we would use the exact message", i.e.,

$$\hat{b}' = \underset{q \in \mathcal{F}}{\operatorname{argmin}} D\big(\frac{\hat{b}}{\hat{t}_i} t_i \,\|\, q\big) \tag{31}$$

where the left argument of the KL-divergence is the belief when we would replace the approximate message $\hat{t}_i$ by the exact message. The updated message $\hat{t}'_i$ that corresponds to this updated belief $\hat{b}'$ is

$$\hat{b}' = \frac{\hat{b}}{\hat{t}_i} \hat{t}'_i \ , \quad \hat{t}'_i = \frac{\hat{t}_i}{\hat{b}} \underset{q \in \mathcal{F}}{\operatorname{argmin}} D\big(\frac{\hat{b}}{\hat{t}_i} t_i \,\|\, q\big) \tag{32}$$

In the case of robot trajectory optimization, the exact messages t_i that arise in typical problems are too complex to be expressed analytically. For instance, when conditioning a collision variable to no-collision, the implied message over the configuration space is extremely complex and, of course, discontinuous. In practice, the trajectory optimization process will be based on a (physical) simulator. In this paper we assume that this simulator can only provide us with the local approximations of the dynamics and costs around a specific configuration x in terms of the system matrices (or vectors) $A_t(x)$, $a_t(x)$, $B_t(x)$, $R_t(x)$, $r_t(x)$. Better simulators could potentially provide us with better, more precise messages that could be employed in an EP framework. Our assumption implies that all messages are Gaussian, as for the LQG case, but the crucial question is which x has been selected, i.e., around which robot configuration we quadratize/linearize the system. We deviate from iLQG which iterates between globally (for all times $t = 0 : T$) linearizing around a current trajectory $x_{0:T}$, and then uses the Ricatti equation to recompute a better trajectory. In our framework we can locally update messages at a certain time step in analogy to EP until we reach local convergence before we move on to another time step. In principle, as with EP, we can freely choose the order in which to update messages including the time steps to focus on.

We first initialize all messages $\mu_{x_{t\text{-}1} \to x_t}$, $\mu_{x_{t+1} \to x_t}$, $\mu_{z_t \to x_t}(x_t)$ for all times t to one, except for $\mu_{x_{-1} \to x_0}(x) = \delta_{x_0 x}$ which encodes conditioning on the known start state. Then, for a certain time step t, we iterate updating all three messages as follows:

(1) Choose a point of approximation $\hat{x}_t$ depending on the mode $\operatorname{argmax}_{x_t} b(x_t)$ of the belief and compute the system matrices $A_t(\hat{x}_t)$, $a_t(\hat{x}_t)$, $B_t(\hat{x}_t)$, $R_t(\hat{x}_t)$, $r_t(\hat{x}_t)$ at this point.

(2) Recompute the messages using this local approximation and equations (20-22).

Algorithm 2 Approximate inference control (AICO)

```
 1: Input: start state $x_0$, control costs $H_{0:T}$, functions
    $A_t(x)$, $a_t(x)$, $B_t(x)$, $R_t(x)$, $r_t(x)$, convergence rate $\alpha$,
    threshold $\theta$
 2: Output: trajectory $x_{0:T}$
 3: initialize $s_0 = x_0$, $S_0^{-1} = 1e10$, $v_{0:T} = 0$, $V_{0:T}^{-1} = 0$,
    $r_{0:T} = 0$, $R_{0:T} = 0$, $k = 0$
 4: repeat
 5:   for $t = 1 : T$ do // forward sweep
 6:     update $s_t$ and $S_t$ using (20)
 7:     if $k = 0$ then
 8:       $\hat{x}_t \leftarrow s_t$
 9:     else
10:       $\hat{x}_t \leftarrow (1-\alpha)\hat{x}_t + \alpha b_t$
11:     end if
12:     access $A_t(\hat{x}_t)$, $a_t(\hat{x}_t)$, $B_t(\hat{x}_t)$, $R_t(\hat{x}_t)$, $r_t(\hat{x}_t)$
13:     update $r_t$ and $R_t$ using (22)
14:     update $v_t$ and $V_t$ using (21)
15:     update $b_t$ and $B_t$ using (19)
16:     if $|\hat{x}_t - b_t|^2 > \theta$ then
17:       $t \leftarrow t - 1$ // repeat this time slice
18:     end if
19:   end for
20:   for $t = T - 1 : 0$ do // backward sweep
21:     ..same updates as above...
22:   end for
23:   $k \leftarrow k + 1$
24: until convergence
```

(3) Compute the current belief over x_t,

$$b(x_t) = \mu_{x_{t\text{-}1} \to x_t}(x_t)\ \mu_{x_{t+1} \to x_t}(x_t)\ \mu_{r_t \to x_t}(x_t)\ .$$

Algorithm 2 is an explicit instantiation of this inference scheme which we call Approximate Inference Control (AICO).

4. Experiments

We test the methods iLQG, AICO, and spline-based gradient optimization on some standard robot motion problems under multiple task constraints. In this section we briefly explain the cost term $c_t(x_t)$ that is implied by these constraints. We assume to have three different kinematic mappings $\phi_i : x \mapsto y_i$ which map the robot state x to different task variables y_i which are the following:

$y_1 \in \mathbb{R}^3$ is the robot's finger tip position. We will constrain this to be close to a goal position for $t = T$.

$y_2 \in \mathbb{R}^2$ is the robot's center of gravity projected on the horizontal plane. We will constrain this to be close to zero throughout the trajectory which implies keeping balance over the foot support.

$y_3 \in \mathbb{R}$ is a scalar measuring collision danger; more precisely, if d_j is the shortest distance between a pair j of collidable objects, then $y_3 = \sum_j \theta(d_j - \epsilon)^2$, with the heavy-side function θ and margin ϵ=0.02 meter.

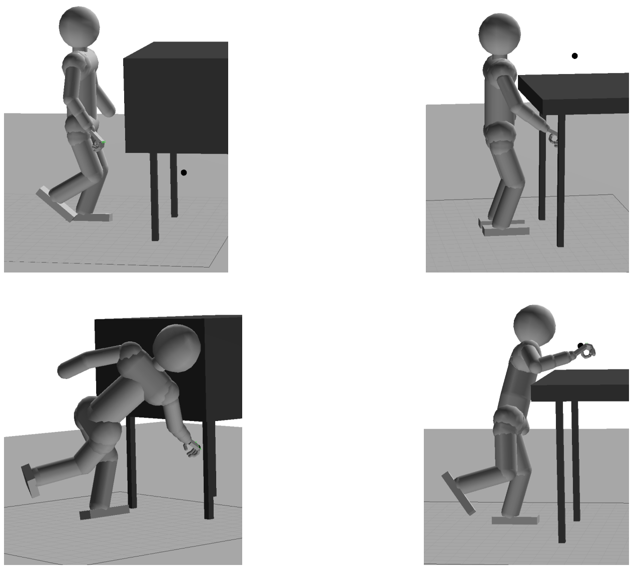

Figure 1. The two reaching problem scenarios. *Top:* Start postures; *bottom:* example final postures.

Table 1. For two reaching problems (1 & 2) and two cost parameter settings (a & b) we instantiated 10 instances by randomizing the target position. We give here the convergence time (=time to reach a cost less than .01 larger than the optimum) and cost with standard deviations.

prob.	method	time (sec)	cost
1a	AICO(α=1,θ=.1)	3.52 $\pm$ 0.70	0.15 $\pm$ 0.03
	AICO(α=.9,θ=.1)	**3.24 $\pm$ 0.52**	0.15 $\pm$ 0.03
	AICO(α=.9,θ=∞)	3.56 $\pm$ 0.61	0.15 $\pm$ 0.03
	iLQG(α=.8)	6.04 $\pm$ 1.14	0.15 $\pm$ 0.03
1b	AICO(α=1,θ=.1)	2.24 $\pm$ 0.51	0.09 $\pm$ 0.01
	AICO(α=.9,θ=.1)	**2.15 $\pm$ 0.51**	0.09 $\pm$ 0.01
	AICO(α=.9,θ=∞)	2.29 $\pm$ 0.57	0.09 $\pm$ 0.01
	iLQG(α=.8)	4.22 $\pm$ 1.79	0.09 $\pm$ 0.01
2a	AICO(α=1,θ=.1)	4.18 $\pm$ 0.52	0.06 $\pm$ 0.01
	AICO(α=.9,θ=.1)	**2.33 $\pm$ 0.65**	0.06 $\pm$ 0.01
	AICO(α=.9,θ=∞)	4.09 $\pm$ 0.97	0.06 $\pm$ 0.01
	iLQG(α=.8)	5.68 $\pm$ 0.92	0.06 $\pm$ 0.01
2b	AICO(α=1,θ=.1)	3.26 $\pm$ 0.51	0.05 $\pm$ 0.00
	AICO(α=.9,θ=.1)	**2.32 $\pm$ 1.20**	0.05 $\pm$ 0.00
	AICO(α=.9,θ=∞)	2.68 $\pm$ 1.19	0.04 $\pm$ 0.00
	iLQG(α=.8)	5.28 $\pm$ 0.88	0.05 $\pm$ 0.01

We will constrain this to be close to zero throughout the trajectory.

Our simulator allows us to compute $\phi_i(x)$ and its Jacobian $J_i(x)$ for any state. We assume we are given targets $y^*_{i,0:T}$ in each task space and (time-dependent) precisions $\varrho_{i,t}$ by which we want to follow the task targets. Given a point of approximation $\hat x_t$ and $\hat J_i = J_i(\hat x_t)$ we define

$$\begin{aligned} c_t(x_t,u_t) &= \sum_{i=1}^3 \varrho_{i,t}[y^*_{i,t} - \phi_i(x_t)]^2 + u_t^\top H_t u_t \\ &\approx \sum_{i=1}^3 \varrho_{i,t}[y^*_{i,t} - \phi_i(\hat x_t) + \hat J_i \hat x_t - \hat J_i x_t]^2 + u_t^\top H_t u_t \\ &= \sum_{i=1}^3 \varrho_{i,t}[x_t^\top \hat J_i^\top \hat J_i x_t - 2(y^*_{i,t} - \phi_i(\hat x_t) + \hat J_i \hat x_t)^\top \hat J_i x_t \\ &\qquad + \text{const}] + u_t^\top H_t u_t \end{aligned} \tag{33}$$

$$R_t = \sum_{i=1}^3 \varrho_{i,t} \hat J_i^\top \hat J_i \tag{34}$$

$$r_t = \sum_{i=1}^3 \varrho_{i,t} \hat J_i^\top (y^*_{i,t} - \phi_i(\hat x_t) + \hat J_i \hat x_t) \tag{35}$$

We investigate simulated humanoid reaching problems in 2 different scenarios. In the first scenario (Figure 1 left), starting from an upright posture the humanoid needs to reach a target point (black dot) with his right index finger which is far below a big obstacle in front of his head. The scenario is interesting in that the multiple criteria all are important: to reach the target without head collision, the robot should early on move its head to the right; and to keep balance on the left foot, the robot needs to stretch its right leg behind and twist the hip a bit. In the second scenario we investigate a more typical reaching problem: with the hand initially below a table the robot needs to move in a wide circle around the table to reach the finger target position. In all experiments we consider a trajectory of length 200. We investigated two sets of precision parameters: In case (a) we chose $\varrho_{1,T} = 10^5$, $\varrho_{1,0:T\text{-}1} = 10^{-4}$ (we require a high precision of 10^5 for the final finger tip position, and practically no endeffector precision during the intermediate trajectory) and $\varrho_{2,0:T} = \varrho_{3,0:T} = 10^5$ (we require even higher precision in the collision and balance variable throughout the trajectory). To investigate the effect of a weak "attracting force" on the endeffector target we also considered parameters $\varrho_{1,T} = 10^2$ in case (b). To collect statistics of the algorithms' performance we randomize the target position in each of 10 trials by adding 3D Gaussian noise with 5cm standard deviation.

Figure 2 displays the optimization curves (i.e., the cost of the current trajectory after each iteration in iLQG, respectively cost of the ML trajectory after a fwd or bwd sweep in AICO) over computation time for both scenarios and parameter settings. We tried also a faster convergence rate $\alpha = 0.9$ for iLQG which lead to divergence of the algorithm in some cases. Generally, we can observe that the inference approach is converging fast – in particular it only needs very few initial iterations to reach good solutions. A trajectory

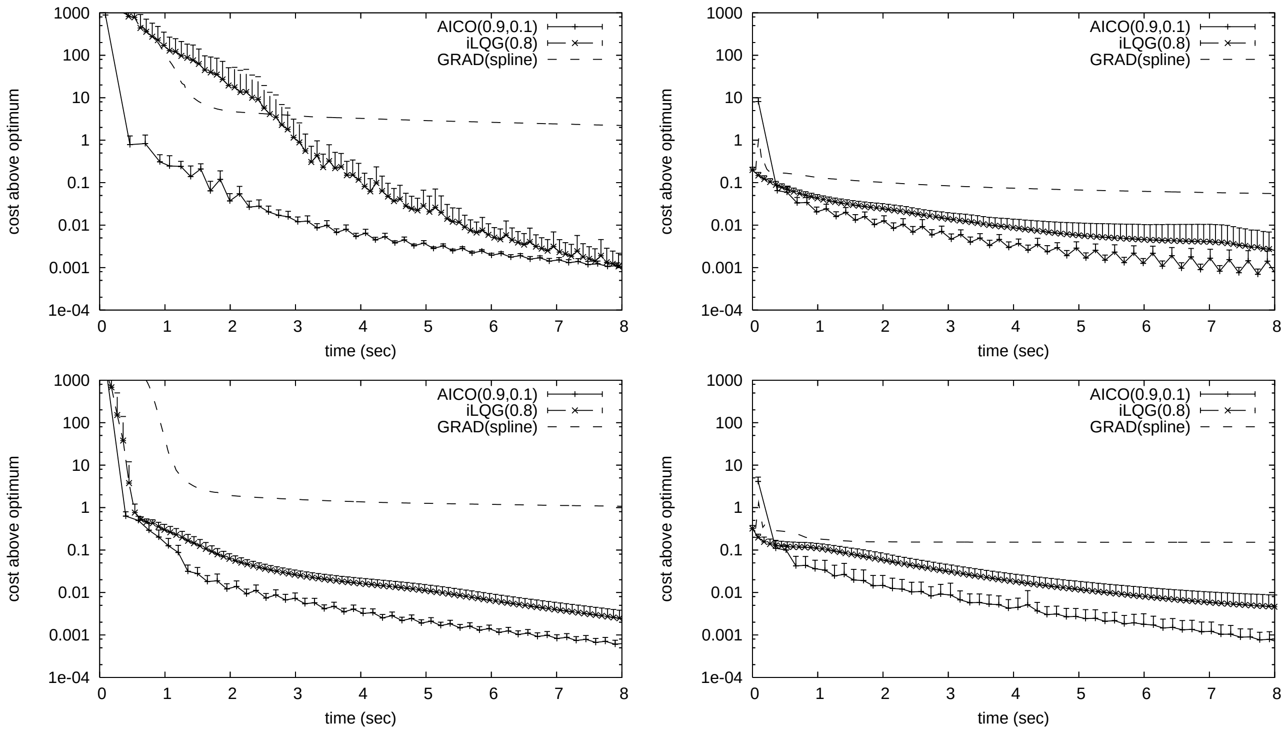

Figure 2. Cost over time during the optimization. *Top:* problem 1a&b; *bottom:* problem 2a&b. Costs are given in log scale and scaled so that zero cost corresponds to the best solution found by any method. Standard deviations are computed over 10 randomized problem instances – for better visibility in the log scaling, only upper errorbars indicating the standard deviation are drawn.

with cost below 0.5 is with high probability collision free. Although each sweep of AICO needs more computation time than an iteration of iLQG, it converges faster measured in computation time. Our interpretation is that the adaptation steps made by AICO when updating a local belief are better estimates of the true local optimum, presumably because also the forward messages (including their uncertainty) are exploited. The gradient-based method, which does not exploit local 2nd order information, performs much worse, also because the spline encoding restricts the space of trajectories.

Table 1 gives results for alternative parameter settings. Here we measure the computation time until a trajectory achieves a cost-above-optimum less than 0.01. AICO consistently performs better than iLQG. The smoothing ($\alpha < 1$) and the repeated update of single time slices ($\theta < \infty$) significantly improve the performance of AICO. iLQG does not converge one some instances for α close to 1.

5. Conclusion

We presented an approach to use approximate inference methods for robot trajectory optimization and stochastic control. Other than previous approaches that tried to generalize Kalman's duality fully to non-LQG cases we aimed for a local approximate solution of the SOC problem and formulated a probabilistic trajectory model for which (1) the ML solution coincides with the deterministic optimal trajectory, and (2) the local Gaussian belief approximation is equivalent to the LQG perturbation model around the optimal trajectory. Based on this model we could derive an approximate message update algorithm in analogy to Expectation Propagation which outperforms previous local approximate SOC algorithms like iLQG.

There are a number of possible extensions of the proposed algorithm. When the simulator returns more detailed information, e.g., on local collision planes, more precise messages (e.g., products of discontinuous boundaries with Gaussians) can be computed in the Expectation Propagation framework. Also non-Gaussian belief approximations in the exponential family can be integrated. Finally, better heuristics for the order of message updates might further increase efficiency.

Acknowledgements

This work was supported by the German Research Foundation, Emmy Noether fellowship TO 409/1-3.

A. Message derivations

Proof. From equation (18) we have

$$\begin{aligned}
&\mu_{x_{t\text{-}1}\to x_t}(x_t)\\
&= \int_{x_{t\text{-}1}} dx_{t\text{-}1}\ P(x_t \mid x_{t\text{-}1})\ \mu_{x_{t\text{-}2}\to x_{t\text{-}1}}(x_{t\text{-}1})\ \mu_{z_{t\text{-}1}\to x_{t\text{-}1}}(x_{t\text{-}1})\\
&= \int_{x_{t\text{-}1}} dx_{t\text{-}1}\ \mathcal{N}(x_t \mid A_{t\text{-}1}x_{t\text{-}1} + a_{t\text{-}1}, Q + B_{t\text{-}1}H^{\text{-}1}B_{t\text{-}1}^\top)\\
&\qquad \cdot \mathcal{N}(x_{t\text{-}1} \mid s_{t\text{-}1}, S_{t\text{-}1})\ \mathcal{N}[x_{t\text{-}1} \mid r_{t\text{-}1}, R_{t\text{-}1}]
\end{aligned}$$

Using the Gaussian product rule the last two terms gives a Gaussian $\mathcal{N}(s_{t\text{-}1} \mid R_{t\text{-}1}^{\text{-}1} r_{t\text{-}1}, S_{t\text{-}1} + R_{t\text{-}1}^{\text{-}1})$ independent of x_t which we can subsume in the normalization. What remains is

$$\begin{aligned}
&\mu_{x_{t\text{-}1}\to x_t}(x_t)\\
&\propto \int_{x_{t\text{-}1}} dx_{t\text{-}1}\ \mathcal{N}(x_t \mid A_{t\text{-}1}x_{t\text{-}1} + a_{t\text{-}1}, Q + B_{t\text{-}1}H^{\text{-}1}B_{t\text{-}1}^\top)\\
&\qquad \cdot \mathcal{N}[x_{t\text{-}1} \mid S_{t\text{-}1}^{\text{-}1}s_{t\text{-}1} + r_{t\text{-}1}, S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1}]\\
&= \int_{x_{t\text{-}1}} dx_{t\text{-}1}\ \mathcal{N}(x_t \mid A_{t\text{-}1}x_{t\text{-}1} + a_{t\text{-}1}, Q + B_{t\text{-}1}H^{\text{-}1}B_{t\text{-}1}^\top)\\
&\qquad \cdot \mathcal{N}(x_{t\text{-}1} \mid (S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1})^{\text{-}1}(S_{t\text{-}1}^{\text{-}1}s_{t\text{-}1} + r_{t\text{-}1}), (S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1})^{\text{-}1})\\
&= \mathcal{N}(x_t \mid A_{t\text{-}1}(S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1})^{\text{-}1}(S_{t\text{-}1}^{\text{-}1}s_{t\text{-}1} + r_{t\text{-}1}) + a_{t\text{-}1}\\
&\qquad , Q + B_{t\text{-}1}H^{\text{-}1}B_{t\text{-}1}^\top + A_{t\text{-}1}(S_{t\text{-}1}^{\text{-}1} + R_{t\text{-}1})^{\text{-}1}A_{t\text{-}1}^\top)
\end{aligned}$$

which gives the messages as in (20). Concerning $\mu_{x_{t+1}\to x_t}(x_t)$ we have,

$$\begin{aligned}
&\mu_{x_{t+1}\to x_t}(x_t)\\
&= \int_{x_{t+1}} dx_{t+1}\ P(x_{t+1} \mid x_t)\ \mu_{x_{t+2}\to x_{t+1}}(x_{t+1})\\
&\qquad \cdot \mu_{z_{t+1}\to x_{t+1}}(x_{t+1})\\
&= \int_{x_{t+1}} dx_{t+1}\ \mathcal{N}(x_{t+1} \mid A_t x_t + a_t, Q + B_t H^{\text{-}1} B_t^\top)\\
&\qquad \cdot \mathcal{N}(x_{t+1} \mid v_{t+1}, V_{t+1})\ \mathcal{N}[x_{t+1} \mid r_{t+1}, R_{t+1}]\\
&\propto \int_{x_{t+1}} dx_{t+1}\ \mathcal{N}(x_{t+1} \mid A_t x_t + a_t, Q + B_t H^{\text{-}1} B_t^\top)\\
&\qquad \cdot \mathcal{N}[x_{t+1} \mid V_{t+1}^{\text{-}1}v_{t+1} + r_{t+1}, V_{t+1}^{\text{-}1} + R_{t+1}]\\
&= \mathcal{N}(A_t x_t + a_t \mid (V_{t+1}^{\text{-}1} + R_{t+1})^{\text{-}1}(V_{t+1}^{\text{-}1}v_{t+1} + r_{t+1})\\
&\qquad , Q + B_t H^{\text{-}1} B_t^\top + (V_{t+1}^{\text{-}1} + R_{t+1})^{\text{-}1})\\
&= \mathcal{N}(x_t \mid -A_t^{\text{-}1}a_t + A_t^{\text{-}1}(V_{t+1}^{\text{-}1} + R_{t+1})^{\text{-}1}(V_{t+1}^{\text{-}1}v_{t+1} + r_{t+1})\\
&\qquad , A_t^{\text{-}1}[Q + B_t H^{\text{-}1} B_t^\top + (V_{t+1}^{\text{-}1} + R_{t+1})^{\text{-}1}]A_t^{\text{-}\top})
\end{aligned}$$

■

References

Attias, H. (2003). Planning by probabilistic inference. *Proc. of the 9th Int. Workshop on Artificial Intelligence and Statistics.*

Bryson, A. E., & Ho, Y.-C. (1969). *Applied optimal control.* Blaisdell Publishing Company.

Chen, Y.-C. (1991). Solving robot trajectory planning problems with uniform cubic b-splines. *Optimal Control Applications and Methods*, *12*, 247–262.

Cooper, G. (1988). A method for using belief networks as influence diagrams. *Proc. of the Fourth Workshop on Uncertainty in Artificial Intelligence* (pp. 55–63).

Dayan, P., & Hinton, G. E. (1997). Using expectation maximization for reinforcement learning. *Neural Computation*, *9*, 271–278.

Kappen, B., Gomez, V., & Opper, M. (2009). Optimal control as a graphical model inference problem.

Minka, T. (2001a). A family of algorithms for approximate bayesian inference. PhD thesis, MIT.

Minka, T. P. (2001b). Expectation propagation for approximate Bayesian inference. *Proc. of the 17th Annual Conf. on Uncertainty in AI (UAI 2001)* (pp. 362–369).

Murphy, K. (2002). Dynamic bayesian networks: Representation, inference and learning. PhD Thesis, UC Berkeley, Computer Science Division.

Schlemmer, M., & Gruebel, G. (1998). Real-time collision- free trajectory optimization of robot manipulators via semi-infinite parameter optimization. *Int. Journal of Robotics Research*, *17*, 1013–1021.

Shachter, R., & Peot (1992). Decision making using probabilistic inference methods. *Proc. of the Eigth Conf. on Uncertainty in Artificial Intelligence* (pp. 276–283).

Shachter, R. D. (1988). Probabilistic inference and influence diagrams. *Operations Research*, *36*, 589–605.

Stengel, R. F. (1986). *Stochastic optimal control.* Wiley.

Todorov, E. (2008). General duality between optimal control and estimation. In *proceedings of the 47th ieee conf. on decision and control.*

Todorov, E., & Li, W. (2005). A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proc. of the american control conference*, 300–306.

Toussaint, M., & Goerick, C. (2007). Probabilistic inference for structured planning in robotics. *Int Conf on Intelligent Robots and Systems (IROS 2007)* (pp. 3068–3073).

Toussaint, M., Harmeling, S., & Storkey, A. (2006). *Probabilistic inference for solving (PO)MDPs* (Technical Report EDI-INF-RR-0934). University of Edinburgh, School of Informatics.

Yedidia, J., Freeman, W., & Weiss, Y. (2001). Understanding belief propagation and its generalizations.

Zhang, J., & Knoll, A. (1995). An enhanced optimization approach for generating smooth robot trajectories in the presence of obstacles. *Proc. of the 1995 European Chinese Automation Conference* (pp. 263–268). London.