Homework 3

Problem 1

Please look through the attached paper to understand the proposed approach, implement the inverted pendulum example in Section VII.A by coding and simulation, and show the state-control trajectories under different scenarios as in Fig. 2.

In proceedings of the American Control Conference 2005

A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems

Emanuel Todorov and Weiwei Li

***Abstract*— We present an iterative Linear-Quadratic-Gaussian method for locally-optimal feedback control of nonlinear stochastic systems subject to control constraints. Previously, similar methods have been restricted to deterministic unconstrained problems with quadratic costs. The new method constructs an affine feedback control law, obtained by minimizing a novel quadratic approximation to the optimal cost-to-go function. Global convergence is guaranteed through a Levenberg-Marquardt method; convergence in the vicinity of a local minimum is quadratic. Performance is illustrated on a limited-torque inverted pendulum problem, as well as a complex biomechanical control problem involving a stochastic model of the human arm, with 10 state dimensions and 6 muscle actuators. A Matlab implementation of the new algorithm is availabe at www.cogsci.ucsd.edu/~todorov.**

I. Introduction

Despite an intense interest in optimal control theory over the last 50 years, solving complex optimal control problems – that do not fit in the well-developed Linear-Quadratic-Gaussian (LQG) formalism – remains a challenge [17]. Most existing numerical methods fall in one of two categories. Global methods based on the Hamilton-Jacobi-Bellman (HJB) equations and the idea of dynamic programming [7], [1] can yield globally-optimal feedback control laws for general stochastic systems. However, such methods involve discretizations of the state and control spaces – which makes them inapplicable to high-dimensional problems, due to the curse of dimensionality. Local methods based on the Maximum Principle [3], [13] avoid the curse of dimensionality, by solving a set of ODEs – via shooting, relaxation, collocation, or gradient descent. But the resulting locally-optimal control laws are open-loop, and stochastic dynamics cannot be taken into account.

An ideal blend of the advantages of local and global methods is provided by Differential Dynamic Programming (DDP) [5]. This method is still local, in the sense that it maintains a representation of a single trajectory and improves it locally. The improvement however does not rely on solving ODEs, but is based on dynamic programming – applied within a "tube" around the current trajectory. DDP is known to have second-order convergence [9], [11], and numerically appears to be more efficient [10] than (efficient implementations of) Newton's method [12]. We recently developed a new method – iterative Linear-Quadratic Regulator design (ILQR) – which is closely related to DDP but turns out to be significantly more efficient: by a factor of 10 on reasonably complex control problems [8]. Our ILQR method uses iterative linearizations of the nonlinear dynamics around the current trajectory, and improves that trajectory via modified Riccati equations. Both DDP and ILQR yield feedback control laws – which is a major advantage compared to open-loop methods. However, these methods are still deterministic. Another shortcoming is that, unlike open-loop methods, DDP and ILQR cannot deal with control constraints and non-quadratic cost functions. The goal of the present paper is to remove these limitations.

While the new algorithm should be applicable to a range of problems, our specific motivation for developing it is the modeling of biological movement. Such modeling has proven extremely useful in the study of how the brain controls movement [15]. Yet, progress has been impeded by the lack of efficient methods that can handle realistic biomechanical control problems. The characteristics of such problems are: high-dimensional nonlinear dynamics; control constraints (e.g. non-negative muscle activations); multiplicative noise, with standard deviation proportional to the control signals [4], [14]; complex performance criteria, that are rarely quadratic in the state variables [6].

Before deriving our new iterative Linear-Quadratic-Gaussian (ILQG) method, we give a more detailed overview of what is new here:

A. Noise

DDP, ILQR, and the new ILQG are dynamic programming methods that use quadratic approximations to the optimal cost-to-go function. All such methods are "blind" to additive noise (see Discussion). However, in many problems of interest the noise is control-dependent, and such noise can easily be captured by quadratic approximations as we show below. Our new ILQG method incorporates control-dependent noise – which turns out to have an effect similar to an energy cost.

B. Constraints

Quadratic approximation methods are presently restricted to unconstrained problems. Generally speaking, constraints make the optimal cost-to-go function non-quadratic. But since we are approximating that function anyway, we might as well take into account the effects of control constraints to the extent possible. Our new ILQG method does that – by

This work is supported by NIH Grant R01-NS045915.

Emanuel Todorov is with the faculty of the Cognitive Science Department, University of California San Diego.

Weiwei Li is with the Deparment of Mechanical and Aerospace Engineering, University of California San Diego.

modifying the feedback gain matrix whenever an element of the open-loop control sequence lies on the constraint boundary.

C. Convexity

Quadratic approximation methods are based on Riccati equations, which are derived by setting up a quadratic optimization problem at time step t, solving it analytically, and obtaining a formula for the optimal cost-to-go function at time step $t-1$. Optimizing a quadratic is only possible when the Hessian is positive-definite. This is of course true in the classic LQG setting, but when LQG methods are used to approximate general nonlinear dynamics with non-quadratic costs, the Hessian can (and in practice does) have zero and even negative eigenvalues. The traditional remedy is to "fix" the Hessian, using a Levenberg-Marquardt method, or an adaptive shift scheme [10], or simply replace it with the identity matrix (which yields the steepest descent method). The problem is that after fixing the Hessian, the optimization at time step t is no longer performed exactly – contrary to what the derivation of Riccati equations assumes. Instead of making this invalid assumption, our new method takes the fixed Hessian into account, and constructs a cost-to-go approximation consistent with the resulting control law. This is done by modified Riccati-like equations.

II. Problem statement

Consider the nonlinear dynamical system described by the stochastic differential equation

$$d\mathbf{x} = \mathbf{f}(\mathbf{x},\mathbf{u})\,dt + F(\mathbf{x},\mathbf{u})\,d\boldsymbol{\omega}$$

with state $\mathbf{x} \in \mathrm{R}^n$, control $\mathbf{u} \in \mathrm{R}^m$, and standard Brownian motion noise $\boldsymbol{\omega} \in \mathrm{R}^p$. Let $\ell(t,\mathbf{x},\mathbf{u}) \geq 0$ be an instantaneous cost rate, $h(\mathbf{x}(T)) \geq 0$ a final cost, T a specified final time, and $\mathbf{u} = \boldsymbol{\pi}(t,\mathbf{x})$ a deterministic control law. Define the cost-to-go function $v^{\boldsymbol{\pi}}(t,\mathbf{x})$ as the total cost expected to accumulate if the system is initialized in state $\mathbf{x}$ at time t, and controlled until time T according to the control law $\boldsymbol{\pi}$:

$$v^{\boldsymbol{\pi}}(t,\mathbf{x}) \triangleq \mathrm{E}\left[h(\mathbf{x}(T)) + \int_t^T \ell(\tau,\mathbf{x}(\tau),\boldsymbol{\pi}(\tau,\mathbf{x}(\tau)))\,d\tau\right]$$

The expectation is taken over the instantiations of the stochastic process $\boldsymbol{\omega}$. The admissible control signals may be constrained: $\mathbf{u}(t) \in \mathcal{U}$, where we assume that $\mathcal{U}$ is convex.

The objective of optimal control is to find the control law $\boldsymbol{\pi}^*$ that minimizes $v^{\boldsymbol{\pi}}(0,\mathbf{x}_0)$. Note that the globally-optimal control law $\boldsymbol{\pi}^*(t,\mathbf{x})$ does not depend on a specific initial state. However, finding this control law in complex problems is unlikely. Instead, we seek locally-optimal control laws: we will approximate $\boldsymbol{\pi}^*$ in the vicinity of the trajectory $\overline{\mathbf{x}}^*(t)$ that results from applying $\boldsymbol{\pi}^*$ to the deterministic system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u})$. Since $\overline{\mathbf{x}}^*$ depends on $\mathbf{x}_0$, so does our approximation to $\boldsymbol{\pi}^*$. Throughout the paper time is discretized as $k = 1 \cdots K$, with time step $\Delta t = T/(K-1)$; thus $\mathbf{u}_k \triangleq \mathbf{u}((k-1)\Delta t)$, and similarly for all other time-varying quantities.

III. Local LQG approximation

The locally-optimal control law is constructed iteratively. Each iteration of the algorithm begins with an open-loop control sequence $\overline{\mathbf{u}}(t)$ and the corresponding "zero-noise" trajectory $\overline{\mathbf{x}}(t)$, obtained by applying $\overline{\mathbf{u}}(t)$ to the deterministic dynamics $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u})$ with $\overline{\mathbf{x}}(0) = \mathbf{x}_0$. This can be done by Euler integration

$$\overline{\mathbf{x}}_{k+1} = \overline{\mathbf{x}}_k + \Delta t\ \mathbf{f}(\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k) \tag{1}$$

or by defining a continuous $\overline{\mathbf{u}}(t)$ via interpolation, applying a continuous-time integrator such as Runge-Kutta, and discretizing the resulting $\overline{\mathbf{x}}(t)$.

Next we linearize the system dynamics and quadratize the cost functions around $\overline{\mathbf{x}}, \overline{\mathbf{u}}$, to obtain a discrete-time linear dynamical system with quadratic cost. Importantly, the linearized dynamics and quadratized cost are expressed not in terms of the actual state and control variables, but in terms of the state and control deviations $\delta\mathbf{x}_k \triangleq \mathbf{x}_k - \overline{\mathbf{x}}_k$, $\delta\mathbf{u}_k \triangleq \mathbf{u}_k - \overline{\mathbf{u}}_k$. Writing the discrete-time dynamics as

$$\delta\mathbf{x}_{k+1} + \overline{\mathbf{x}}_{k+1} = \delta\mathbf{x}_k + \overline{\mathbf{x}}_k + \Delta t\ \mathbf{f}(\delta\mathbf{x}_k + \overline{\mathbf{x}}_k, \delta\mathbf{u}_k + \overline{\mathbf{u}}_k)$$

expanding $\mathbf{f}$ around $\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k$ up to first order, and subtracting (1), our original optimal control problem is approximated locally by the following modified LQG problem:

$$\begin{aligned}
\delta\mathbf{x}_{k+1} &= A_k\delta\mathbf{x}_k + B_k\delta\mathbf{u}_k + \mathcal{C}_k(\delta\mathbf{u}_k)\,\boldsymbol{\xi}_k \\
\mathcal{C}_k(\delta\mathbf{u}_k) &\triangleq [\mathbf{c}_{1,k} + C_{1,k}\delta\mathbf{u}_k \ \cdots\ \mathbf{c}_{p,k} + C_{p,k}\delta\mathbf{u}_k] \\
\mathrm{cost}_k &= q_k + \delta\mathbf{x}_k^{\mathsf{T}}\mathbf{q}_k + \frac{1}{2}\delta\mathbf{x}_k^{\mathsf{T}}Q_k\delta\mathbf{x}_k \\
&\quad + \delta\mathbf{u}_k^{\mathsf{T}}\mathbf{r}_k + \frac{1}{2}\delta\mathbf{u}_k^{\mathsf{T}}R_k\delta\mathbf{u}_k + \delta\mathbf{u}_k^{\mathsf{T}}P_k\delta\mathbf{x}_k
\end{aligned} \tag{2}$$

where $\delta\mathbf{x}_1 = 0$, $\boldsymbol{\xi}_k \sim \mathrm{N}(0;\mathrm{I}_p)$, and the last time step is K. The quantities that define this modified LQG problem are $A_k, B_k, \mathbf{c}_{i,k}, C_{i,k}, q_k, \mathbf{q}_k, Q_k, \mathbf{r}_k, R_k, P_k, K$. At each $(\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k)$, these quantities are obtained from the original problem as:

$$\begin{aligned}
A_k &= \mathrm{I} + \Delta t\ \partial\mathbf{f}/\partial\mathbf{x}; \quad B_k = \Delta t\ \partial\mathbf{f}/\partial\mathbf{u} \\
\mathbf{c}_{i,k} &= \sqrt{\Delta t}\ F^{[i]}; \quad C_{i,k} = \sqrt{\Delta t}\ \partial F^{[i]}/\partial\mathbf{u} \\
q_k &= \Delta t\ \ell; \quad \mathbf{q}_k = \Delta t\ \partial\ell/\partial\mathbf{x} \\
Q_k &= \Delta t\ \partial^2\ell/\partial\mathbf{x}\partial\mathbf{x}; \quad P_k = \Delta t\ \partial^2\ell/\partial\mathbf{u}\partial\mathbf{x} \\
\mathbf{r}_k &= \Delta t\ \partial\ell/\partial\mathbf{u}; \quad R_k = \Delta t\ \partial^2\ell/\partial\mathbf{u}\partial\mathbf{u}
\end{aligned} \tag{3}$$

where $F^{[i]}$ denotes the i^{th} column of the matrix F. At the final time step $k = K$ the cost model is $q_K = h$; $\mathbf{q}_K = \partial h/\partial\mathbf{x}$; $Q_K = \partial^2 h/\partial\mathbf{x}\partial\mathbf{x}$, and the control-related cost terms $\mathbf{r}_K, R_K, P_K$ do not affect the optimization. The $\sqrt{\Delta t}$ term appears because the covariance of Brownian motion grows linearly with time. The i^{th} column of the matrix $\mathcal{C}_k(\delta\mathbf{u}_k)$ is $\mathbf{c}_i + C_i\delta\mathbf{u}_k$. Thus the noise covariance is

$$\mathrm{Cov}\left[\mathcal{C}_k(\delta\mathbf{u}_k)\,\boldsymbol{\xi}_k\right] = \sum_{i=1}^{p}(\mathbf{c}_{i,k} + C_{i,k}\delta\mathbf{u}_k)(\mathbf{c}_{i,k} + C_{i,k}\delta\mathbf{u}_k)^{\mathsf{T}}$$

Note that we are using a simplified noise model, where $F(\mathbf{x},\mathbf{u})$ is only linearized with respect to $\delta\mathbf{u}$. This is

sufficient to capture noise that is multiplicative in the control signal – which is what we are mainly interested in. It is straightforward to include state-dependent noise as well, and repeat the derivation that follows.

Now that we have an LQG approximation to our original optimal control problem, we can proceed with computing an approximately-optimal control law. This is done in the following two sections. We then describe the main iteration of the algorithm, which constructs a convergent sequence of LQG approximations.

IV. Computing the cost-to-go function

The approximately-optimal control law for the LQG approximation will be affine, in the form $\delta\mathbf{u} = \boldsymbol{\pi}_k(\delta\mathbf{x}) = \mathbf{l}_k + L_k\delta\mathbf{x}$. The unusual open-loop component $\mathbf{l}_k$ (letter 'el') arises because we are dealing with state and control deviations, and is needed to make the algorithm iterative (see below). The control law is approximately-optimal because we may have control constraints and non-convex costs.

Suppose the control law $\boldsymbol{\pi}$ has already been designed for time steps $k \cdots K-1$. Then the cost-to-go function $v_k(\delta\mathbf{x})$ is well-defined – as the cost expected to accumulate if system (2) is initialized in state $\delta\mathbf{x}$ at time step k, and controlled according to $\boldsymbol{\pi}$ for the remaining time steps. We will show by induction (backwards in time) that if the control law is affine, the corresponding cost-to-go function remains in the quadratic form

$$v_k(\delta\mathbf{x}) = s_k + \delta\mathbf{x}^{\mathsf{T}}\mathbf{s}_k + \tfrac{1}{2}\delta\mathbf{x}^{\mathsf{T}}S_k\delta\mathbf{x} \tag{4}$$

for all k. At the last time step this holds, because the final cost is a quadratic that does not depend on the control signal. Now suppose that v is in the above form for time steps $k+1 \cdots K$. Using the shortcut $\boldsymbol{\pi}$ in place of the control signal $\mathbf{l}_k + L_k\delta\mathbf{x}$ that our control law generates, the Bellman equation for the cost-to-go function is

$$\begin{aligned} v_k(\delta\mathbf{x}) &= \text{immediate cost } + \mathrm{E}\left[v_{k+1}(\text{next state})\right] \\ &= q_k + \delta\mathbf{x}^{\mathsf{T}}\left(\mathbf{q}_k + \tfrac{1}{2}Q_k\delta\mathbf{x}\right) + \boldsymbol{\pi}^{\mathsf{T}}\left(\mathbf{r}_k + \tfrac{1}{2}R_k\boldsymbol{\pi}\right) \\ &\quad + \boldsymbol{\pi}^{\mathsf{T}}P_k\delta\mathbf{x} + \mathrm{E}\left[v_{k+1}\left(A_k\delta\mathbf{x} + B_k\boldsymbol{\pi} + \mathcal{C}_k\boldsymbol{\xi}_k\right)\right] \end{aligned}$$

Evaluating the expectation term $\mathrm{E}[\cdot]$ above yields

$$\begin{aligned} & s_{k+1} + \left(A_k\delta\mathbf{x} + B_k\boldsymbol{\pi}\right)^{\mathsf{T}}\mathbf{s}_{k+1} + \\ & \tfrac{1}{2}\left(A_k\delta\mathbf{x} + B_k\boldsymbol{\pi}\right)^{\mathsf{T}}S_{k+1}\left(A_k\delta\mathbf{x} + B_k\boldsymbol{\pi}\right) + \\ & \tfrac{1}{2}\ \mathrm{trace}\left(\textstyle\sum_{i=1}^{p}\left(\mathbf{c}_{i,k} + C_{i,k}\boldsymbol{\pi}\right)\left(\mathbf{c}_{i,k} + C_{i,k}\boldsymbol{\pi}\right)^{\mathsf{T}}S_{k+1}\right) \end{aligned}$$

Using the fact that $\mathrm{trace}(UV) = \mathrm{trace}(VU)$, the $\frac{1}{2}\ \mathrm{trace}(\cdot)$ term above becomes

$$\begin{aligned} & \tfrac{1}{2}\boldsymbol{\pi}^{\mathsf{T}}\left(\textstyle\sum_i C_{i,k}^{\mathsf{T}}S_{k+1}C_{i,k}\right)\boldsymbol{\pi} + \\ & \boldsymbol{\pi}^{\mathsf{T}}\left(\textstyle\sum_i C_{i,k}^{\mathsf{T}}S_{k+1}\mathbf{c}_{i,k}\right) + \tfrac{1}{2}\left(\textstyle\sum_i \mathbf{c}_{i,k}^{\mathsf{T}}S_{k+1}\mathbf{c}_{i,k}\right) \end{aligned}$$

Combining the results and grouping terms, the cost-to-go is

$$\begin{aligned} v_k(\delta\mathbf{x}) &= q_k + s_{k+1} + \tfrac{1}{2}\textstyle\sum_i \mathbf{c}_i^{\mathsf{T}}S_{k+1}\mathbf{c}_i \\ &\quad + \delta\mathbf{x}^{\mathsf{T}}\left(\mathbf{q}_k + A_k^{\mathsf{T}}\mathbf{s}_{k+1}\right) \\ &\quad + \tfrac{1}{2}\delta\mathbf{x}^{\mathsf{T}}\left(Q_k + A_k^{\mathsf{T}}S_{k+1}A_k\right)\delta\mathbf{x} \\ &\quad + \boldsymbol{\pi}^{\mathsf{T}}\left(\mathbf{g} + G\delta\mathbf{x}\right) + \tfrac{1}{2}\boldsymbol{\pi}^{\mathsf{T}}H\boldsymbol{\pi} \end{aligned} \tag{5}$$

The shortcuts $\mathbf{g}, G, H$ appearing on the last line of (5) are defined at each time step as

$$\begin{aligned} \mathbf{g} &\triangleq \mathbf{r}_k + B_k^{\mathsf{T}}\mathbf{s}_{k+1} + \textstyle\sum_i C_{i,k}^{\mathsf{T}}S_{k+1}\mathbf{c}_{i,k} \\ G &\triangleq P_k + B_k^{\mathsf{T}}S_{k+1}A_k \\ H &\triangleq R_k + B_k^{\mathsf{T}}S_{k+1}B_k + \textstyle\sum_i C_{i,k}^{\mathsf{T}}S_{k+1}C_{i,k} \end{aligned} \tag{6}$$

At this point one may notice that the expression for $v_k(\delta\mathbf{x})$ is a quadratic function of $\boldsymbol{\pi}$, and set the control signal to the value of $\boldsymbol{\pi}$ which makes the gradient vanish: $\delta\mathbf{u}_k = -H^{-1}\mathbf{g}_k - H^{-1}G\delta\mathbf{x}$. But we will not assume this specific form of the control law here, because H may have negative eigenvalues (in which case the above $\delta\mathbf{u}$ is not a minimum), and also because some control constraints may be violated. Instead we will defer the computation of the control law to the next section. All we assume for now is that the control law computed later will be in the general form $\delta\mathbf{u} = \mathbf{l}_k + L_k\delta\mathbf{x}$. With this assumption we can complete the computation of the cost-to-go function. Replacing $\boldsymbol{\pi}$ with $\mathbf{l}_k + L_k\delta\mathbf{x}$, and noting that the square matrix S_k is symmetric, the $\boldsymbol{\pi}$-dependent expression in the last line of (5) becomes

$$\begin{aligned} & \mathbf{l}_k^{\mathsf{T}}\mathbf{g} + \tfrac{1}{2}\mathbf{l}_k^{\mathsf{T}}H\mathbf{l}_k + \delta\mathbf{x}^{\mathsf{T}}\left(G^{\mathsf{T}}\mathbf{l}_k + L_k^{\mathsf{T}}\mathbf{g} + L_k^{\mathsf{T}}H\mathbf{l}_k\right) \\ & + \tfrac{1}{2}\delta\mathbf{x}^{\mathsf{T}}\left(L_k^{\mathsf{T}}HL_k + L_k^{\mathsf{T}}G + G^{\mathsf{T}}L_k\right)\delta\mathbf{x} \end{aligned}$$

We now see that the cost-to-go function remains quadratic in $\delta\mathbf{x}$, which completes the induction proof. Thus we have

Lemma (ILQG) For any affine control law in the form $\delta\mathbf{u}(k, \delta\mathbf{x}) = \mathbf{l}_k + L_k\delta\mathbf{x}$, the cost-to-go for problem (2) is in the form (4) for all k. At the final time step K, the cost-to-go parameters are $S_K = Q_K,\ \mathbf{s}_K = \mathbf{q}_K,\ s_K = q_K$. For $k < K$ the parameters can be computed recursively as

$$\begin{aligned} S_k &= Q_k + A_k^{\mathsf{T}}S_{k+1}A_k + L_k^{\mathsf{T}}HL_k + L_k^{\mathsf{T}}G + G^{\mathsf{T}}L_k \\ \mathbf{s}_k &= \mathbf{q}_k + A_k^{\mathsf{T}}\mathbf{s}_{k+1} + L_k^{\mathsf{T}}H\mathbf{l}_k + L_k^{\mathsf{T}}\mathbf{g} + G^{\mathsf{T}}\mathbf{l}_k \\ s_k &= q_k + s_{k+1} + \tfrac{1}{2}\textstyle\sum_i \mathbf{c}_{i,k}^{\mathsf{T}}S_{k+1}\mathbf{c}_{i,k} + \tfrac{1}{2}\mathbf{l}_k^{\mathsf{T}}H\mathbf{l}_k + \mathbf{l}_k^{\mathsf{T}}\mathbf{g} \end{aligned} \tag{7}$$

where $\mathbf{g}, G, H$ are defined in (6). The total cost is s_1. ■

If we are not concerned with negative eigenvalues of H or violations of control constraints, and set $\mathbf{l}_k = -H^{-1}\mathbf{g}$, $L_k = -H^{-1}G$ as mentioned above, a number of terms cancel and (7) reduces to

$$\begin{aligned} S_k &= Q_k + A_k^{\mathsf{T}}S_{k+1}A_k - G^{\mathsf{T}}H^{-1}G \\ \mathbf{s}_k &= \mathbf{q}_k + A_k^{\mathsf{T}}\mathbf{s}_{k+1} - G^{\mathsf{T}}H^{-1}\mathbf{g} \\ s_k &= q_k + s_{k+1} + \tfrac{1}{2}\textstyle\sum_i \mathbf{c}_{i,k}^{\mathsf{T}}S_{k+1}\mathbf{c}_{i,k} - \tfrac{1}{2}\mathbf{g}^{\mathsf{T}}H^{-1}\mathbf{g} \end{aligned}$$

If we further remove the control-dependent noise (by setting $C_{i,k} = 0$) and the linear terms in the cost function (by setting $\mathbf{q}_k = \mathbf{r}_k = 0$), we see that $\mathbf{g} = \mathbf{l}_k = \mathbf{s}_k = 0$ and the first line of (7) reduces to the familiar LQR discrete-time Riccati equation

$$\begin{aligned} S_k &= Q_k + A_k^{\mathsf{T}}S_{k+1}A_k \\ &\quad - A_k^{\mathsf{T}}S_{k+1}B_k\left(R_k + B_k^{\mathsf{T}}S_{k+1}B_k\right)^{-1}B_k^{\mathsf{T}}S_{k+1}A_k \end{aligned}$$

Thus our method can be reduced to familiar methods in special cases, but has the added flexibility of keeping the quadratic cost-to-go calculation consistent regardless of how the affine feedback control law is computed.

It can be seen from the above equations that in the absence of control-dependent noise (i.e. when $C = 0$), additive noise has no effect on the control law. However, when $C > 0$, increasing the additive noise magnitude $\mathbf{c}$ results in changes of the control law.

V. Computing the control law

As we saw in (5), the cost-to-go function $v_k(\delta\mathbf{x})$ depends on the control $\delta\mathbf{u}_k = \boldsymbol{\pi}_k(\delta\mathbf{x})$ through the term

$$a(\delta\mathbf{u},\delta\mathbf{x}) = \delta\mathbf{u}^{\mathsf{T}}(\mathbf{g} + G\delta\mathbf{x}) + \tfrac{1}{2}\delta\mathbf{u}^{\mathsf{T}} H \delta\mathbf{u} \qquad (8)$$

where we have suppressed the time index k. Ideally we would choose the $\delta\mathbf{u}$ that minimizes a for every $\delta\mathbf{x}$, subject to whatever control constraints are present. However, this is not always possible within the family of affine control laws $\delta\mathbf{u} = \mathbf{l} + L\delta\mathbf{x}$ that we are considering. Since the goal of the LQG stage is to approximate the optimal controller for the nonlinear system in the vicinity of $\overline{\mathbf{x}}$, we will give preference to linear control laws that are optimal/feasible for small $\delta\mathbf{x}$, even if that (unavoidably) makes them sub-optimal/infeasible for larger $\delta\mathbf{x}$. In particular we need to make sure that for $\delta\mathbf{x} = 0$, the new open-loop control $\delta\mathbf{u} = \mathbf{l}$ performs no worse than the current open-loop control $\delta\mathbf{u} = 0$. Since $a(0,\delta\mathbf{x}) = 0$, this holds if $a(\mathbf{l},0) = \mathbf{l}^{\mathsf{T}}\mathbf{g} + \frac{1}{2}\mathbf{l}^{\mathsf{T}} H \mathbf{l} \le 0$. The latter is always achievable by setting $\mathbf{l} = -\epsilon\mathbf{g}$ for a small enough $\epsilon \ge 0$.

A. First-order and second-order methods

The gradient $\nabla_{\delta\mathbf{u}} a(\delta\mathbf{u},\delta\mathbf{x})$ evaluated at $\delta\mathbf{u} = 0$ is $\mathbf{g} + G\delta\mathbf{x}$. Therefore we can make an improvement along the gradient of a by setting $\delta\mathbf{u} = -\epsilon(\mathbf{g} + G\delta\mathbf{x})$. If we are only interested in open-loop control, we can use $\delta\mathbf{u} = -\epsilon\mathbf{g}$. If the new control signal (for the nonlinear system) $\overline{\mathbf{u}} - \epsilon\mathbf{g}$ violates the constraints, we have to reduce ϵ until the constraints are satisfied. Note that $\overline{\mathbf{u}}$ is by definition a feasible control signal, so unless it lies on the constraint boundary (and $\mathbf{g}$ points inside the feasible set) we can find an $\epsilon > 0$ for which $\overline{\mathbf{u}} - \epsilon\mathbf{g}$ is also feasible. This gives a first-order method.

To obtain a second-order method we use the Hessian H. If H is positive semi-definite we can compute the unconstrained optimal control law $\delta\mathbf{u} = -H^{-1}(\mathbf{g} + G\delta\mathbf{x})$, and deal with the control constraints as described below. But when H has negative eigenvalues, there exist $\delta\mathbf{u}$'s that make a (and therefore v) arbitrarily negative. Note that the cost-to-go function for the nonlinear problem is always non-negative, but since we are using an approximation to the true cost we may (and in practice do) encounter situations where a does not have a minimum. In that case the gradient $\nabla_{\delta\mathbf{u}} a = \mathbf{g} + G\delta\mathbf{x}$ is still correct, and so the true cost-to-go decreases in the direction $-\mathcal{H}^{-1}(\mathbf{g} + G\delta\mathbf{x})$ for any positive definite matrix $\mathcal{H}$. Thus we use a matrix $\mathcal{H}$ that "resembles" H, but is positive definite. We have experimented with several choices of $\mathcal{H}$, and found the best convergence using a method related to Levenberg-Marquardt: (1) compute the eigenvalue decomposition $[V, D] = \mathrm{eig}(H)$; (2) replace all negative elements of the diagonal matrix D with 0; (3) add a positive constant λ to the diagonal of D; (4) set $\mathcal{H} = VDV^{\mathsf{T}}$, using the modified D from steps (2) and (3). The constant λ is set and adapted in the main iteration of the algorithm, as described below. Note that $\mathcal{H}^{-1}$ can be efficiently computed as $VD^{-1}V^{\mathsf{T}}$.

B. Constrained second-order methods

The problem here is to find the affine control law $\delta\mathbf{u} = \mathbf{l} + L\delta\mathbf{x}$ minimizing (8) subject to constraints $\delta\mathbf{u} + \overline{\mathbf{u}} \in \mathcal{U}$, assuming that H has already been replaced with a positive definite $\mathcal{H}$. We first optimize the open-loop component $\mathbf{l}$ for $\delta\mathbf{x} = 0$, and then deal with the feedback term $L\delta\mathbf{x}$. The unconstrained minimum is $\delta\mathbf{u}^* = -\mathcal{H}^{-1}\mathbf{g}$. If it satisfies $\delta\mathbf{u}^* + \overline{\mathbf{u}} \in \mathcal{U}$ we are done. Otherwise we have two options. The more efficient but less accurate method is to backtrack once, i.e. to find the maximal $\epsilon \in [0;1]$ such that $\epsilon\delta\mathbf{u}^* + \overline{\mathbf{u}} \in \mathcal{U}$. This is appropriate in the early phase of the iterative algorithm when $\overline{\mathbf{x}}$ is still far away from $\overline{\mathbf{x}}^*$; in that phase it makes more sense to quickly improve the control law rather than refine the solution to an LQG problem that is an inaccurate approximation to the original problem. But in the final phase of the iterative algorithm we want to obtain the best control law possible for the given LQG problem. In that phase we use quadratic programming.

Once $\mathbf{l}$ is determined, we have to compute the feedback gain matrix L. Given that $\delta\mathbf{x}$ is unconstrained, the only general way to enforce the constraints $\mathcal{U}$ is to set $L = 0$. In practice we do not want to be that conservative, since we are looking for an approximation to the nonlinear problem that is valid around $\delta\mathbf{x} = 0$. If $\mathbf{l} + \overline{\mathbf{u}}$ is inside $\mathcal{U}$, small changes $L\delta\mathbf{x}$ will not cause constraint violations and so we can use the optimal $L = -\mathcal{H}^{-1}G$. But if $\mathbf{l} + \overline{\mathbf{u}}$ lies on the constraint boundary $\partial\mathcal{U}$, we have to modify L so that $L\delta\mathbf{x}$ can only cause changes along the boundary. Modifying L is straightforward in the typical case when the range of each element of $\mathbf{u}$ is specified independently. In that case we simply set to 0 the rows of $-\mathcal{H}^{-1}G$ corresponding to elements of $\mathbf{l} + \overline{\mathbf{u}}$ that have reached their limits.

VI. Main iteration

Each iteration of ILQG starts with an open-loop control sequence $\overline{\mathbf{u}}^{(i)}(t)$, and a corresponding state trajectory $\overline{\mathbf{x}}^{(i)}(t)$ computed as in (1). As described in the previous sections, we then build a local LQG approximation around $\overline{\mathbf{x}}^{(i)}, \overline{\mathbf{u}}^{(i)}$ and design an affine control law for the linearized system, in the form $\delta\mathbf{u}_k = \mathbf{l}_k + L_k\delta\mathbf{x}_k$. This control law is applied forward in time to the linearized system $\delta\mathbf{x}_{k+1} = A_k\delta\mathbf{x}_k + B_k\delta\mathbf{u}_k$, initialized at $\delta\mathbf{x}_1 = 0$. The new open-loop controls $\widetilde{\mathbf{u}}_k = \overline{\mathbf{u}}_k^{(i)} + \delta\mathbf{u}_k$ are computed along the way, enforcing $\widetilde{\mathbf{u}} \in \mathcal{U}$ if necessary. If the sequences $\overline{\mathbf{u}}^{(i)}$ and $\widetilde{\mathbf{u}}$ are sufficiently close, the iteration ends. Note that in the absence of the $\mathbf{l}_k$ term we would have $\delta\mathbf{u}_k = 0$ and

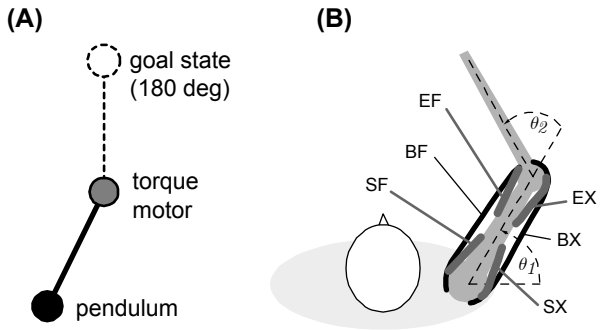

Fig. 1. Illustration of the control problems being studied. In (B), we have elbow flexors (EF), elbow extensors (EX), shoulder flexors (SF), shoulder extensors (SX), biarticulate flexors (BF), and biarticulate extensors (BX).

$\delta \mathbf{x}_k = 0$ for all k, and so the reference trajectory used to center the LQG approximation will never improve.

If $\widetilde{\mathbf{u}}$ results in better performance than $\overline{\mathbf{u}}^{(i)}$, we set $\overline{\mathbf{u}}^{(i+1)} = \widetilde{\mathbf{u}}$ and decrease the Levenberg-Marquardt constant λ. Otherwise we increase λ and recompute $\widetilde{\mathbf{u}}$. When $\lambda = 0$ we have a Newton method using the true Hessian. When λ is large the Hessian is effectively replaced by $\lambda \mathrm{I}$, and so the algorithm takes very small steps in the direction of the gradient. We have found empirically that using such an adaptive method make a difference – in terms of both robustness and speed of convergence.

VII. Optimal control problems to be studied

We have thus far tested the method on two problems, both of which have nonlinear dynamics, non-quadratic costs, control constraints, and (for the second problem) multiplicative noise.

A. An inverted pendulum

We use the popular inverted pendulum problem (Fig 1A), with a limit on the torque that the motor can generate, and also a quadratic cost on the torque. Thus the optimal solutions are not always in the form of bang-bang control (which is the case when the control cost is absent) but exhibit both torque saturation and continuous transitions between torque limits. The dynamics are

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u - 4 \sin x_1 \end{aligned}$$

where the state variables are $x_1 = \theta$, $x_2 = \dot{\theta}$. The goal is to make the pendulum swing up (corresponding to a 180 deg angle) and also make it stop – at the final time T. The control objective is to find the control $u(t)$ that minimizes the performance index

$$J_0 = (1 + \cos x_1(T))^2 + 0.1\ x_2\,(T)^2 + 0.01 \int_0^T u\,(t)^2\, dt$$

We use a time step of 10 msec, $T = 4$sec, and the maximum control torque that can be generated is $|u| \leq 2$.

B. A model of the human arm

The second model we study is rather complex, and we lack the space to describe it in detail – see [8]. We model the nonlinear dynamics of a 2-link 6-muscle human arm moving in the horizontal plane (Fig 1B), using standard equations of motion in the form

$$\mathcal{M}(\theta)\ddot{\theta} + \mathcal{C}(\theta, \dot{\theta}) + \mathcal{B}\dot{\theta} = \tau,$$

where $\theta \in R^2$ is the joint angle vector (shoulder: θ_1, elbow: θ_2), $\mathcal{M}(\theta) \in R^{2\times 2}$ is the inertia matrix, $\mathcal{C}(\theta, \dot{\theta}) \in R^2$ is the vector of centripetal and Coriolis forces, $\mathcal{B}$ is the joint friction matrix, and $\tau \in R^2$ is the joint torque that the muscles generate. The muscles are partitioned into 6 actuator groups. The joint torques produced by a muscle are a function of its posture-dependent moment arm, length-velocity-tension curve, and activation level. Muscles act like first-order nonlinear low-pass filters, and thus have activations states a. Each muscle receives control u_i that is polluted with multiplicative noise, and has activation dynamics $\dot{a}_i = (u_i - a_i)/t(u_i, a_i)$. Thus the state $\mathbf{x} = \left[\theta_1; \dot{\theta}_2; \theta_1; \dot{\theta}_2; a_1; a_2; a_3; a_4; a_5; a_6\right]$ is 10-dimensional.

The task we study is reaching: the arm has to start at some initial position and move to a target in a specified time interval. It also has to stop at the target, and do all that with minimal energy consumption. There are good reasons to believe that such costs are indeed relevant to the neural control of movement [14]. The cost function is defined as

$$\begin{aligned} J_0 &= \left\|\mathbf{e}\left(\theta(T)\right) - \mathbf{e}^*\right\|^2 + 0.001\ \left\|\dot{\mathbf{e}}\left(\theta(T), \dot{\theta}(T)\right)\right\|^2 \\ &\quad + \frac{1}{2}\int_0^T 0.0001\,\|\mathbf{u}\|^2\ dt \end{aligned}$$

where $\mathbf{e}\,(\theta)$ is the forward kinematics transformation from joint coordinates to end-point coordinates, and the target $\mathbf{e}^*$ is defined in end-point coordinates.

VIII. Numerical results

Since the pendulum has a two-dimensional state space, we can discretize it with a dense grid and solve the time-varying Hamilton-Jacobi-Bellman PDE numerically. We used a 100x100 grid, and 400 time step (4 sec interval, 10 msec time step).

Fig 2 shows the optimal trajectories of the pendulum, according to the HJB solution, in gray. The best solutions found by our iterative method, after 3 restarts (with constant initial control sequences $u\,(t) = 0, 1, -1$) are shown in black. Note the close correspondence. In the only case where we saw a mismatch, running the algorithm with additional initial conditions found a better local minimum (dotted line) that agrees with the HJB solution.

We also applied ILQG to the human arm model described above. Note that this model is stochastic: we include multiplicative noise in the muscle activations, with standard deviation equal to 20% of the control signal. Fig 3 shows average behavior: hand paths in (A), tangential speed profiles

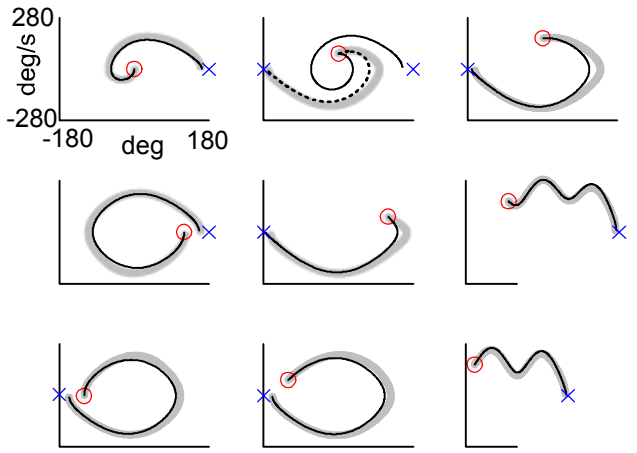

Fig. 2. State-control trajectories for the inverted pendulum, found by ILQG (black) and a global method (gray). Circles mark the starting state, crosses mark the target (repeated every 360 deg).

in (B), and muscle activations in (C). Both the movement kinematics and the muscle activations share many features with experimental data on human arm movements – but a detailed discussion of the relevance to Motor Control is beyond the scope of this paper.

Fig 4 illustrates the robustness to noise: open-loop control in (A), closed-loop control in (B), and closed-loop control optimized for a deterministic system in (C). Closed-loop control is based on the time-varying feedback gain matrix L generated by the ILQG method, while open-loop control only uses the final $\overline{\mathbf{u}}$ constructed by the algorithm. As the endpoint error ellipses show, the feedback control scheme substantially reduces the effects of the noise, and benefits from being optimized for the correct multiplicative noise model.

Another encouraging result is that in terms of CPU time, the complex arm model does not require much more computation than the pendulum. Fig 5A shows how the total cost (for the arm control problem) decreased over iterations of the algorithm, for reaching in 8 different directions. On average, ILQG found a locally-optimal time-varying feedback control law in about 5 seconds (on a 3.2GHz Pentium 4 machine, in Matlab). It should be noted that while the pendulum dynamics could be linearized analytically, the arm dynamics is too complex and so we had to resort to a centered finite difference approximation (which turned out to be the computational bottleneck).

Finally, we explored the issue of local minima for the arm control problem (Fig 5B). We initialized the open-loop controls to random sequences, whose elements were uniformly sampled between 0 and 0.1 (the allowed range of muscle control signals was between 0 and 1). The resulting initial trajectories are shown as an inset in Fig 5B. Note that they all go in roughly the same direction, because some arm muscle are stronger than others. We used 10 different initialization, for each of 8 movement directions.

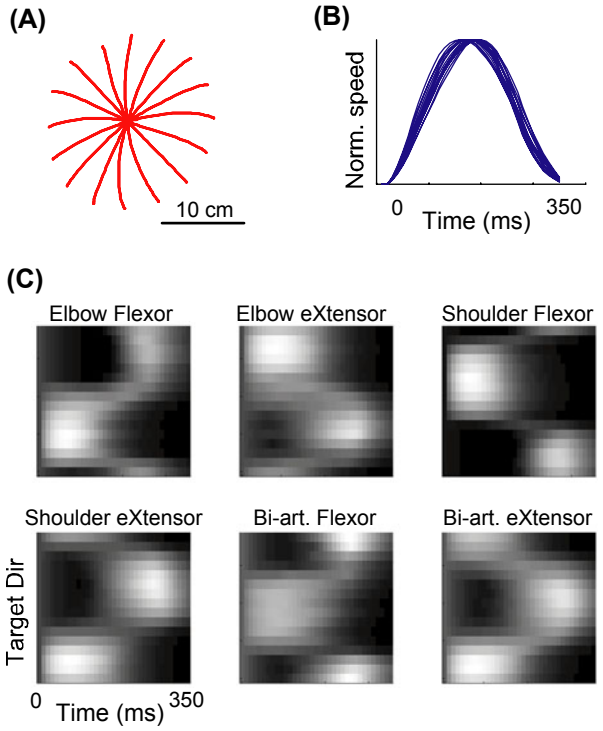

Fig. 3. Average behavior of the ILQG controller for reaching movements, using a 2-link 6-muscle model of the human arm. (A) Hand paths for movement in 16 directions; (B) Speed profiles; (C) Muscle activations.

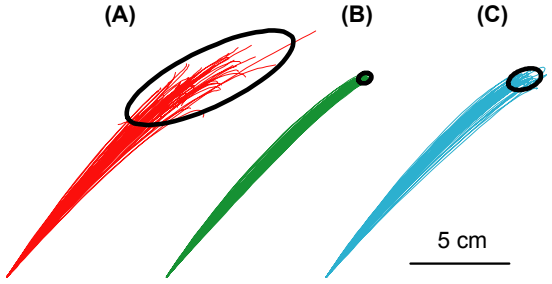

Fig. 4. Effects of control-dependent noise on hand reaching trajectories, under different control laws. (A) open-loop control; (B) closed-loop control; (C) closed-loop controller optimized for deterministic system.

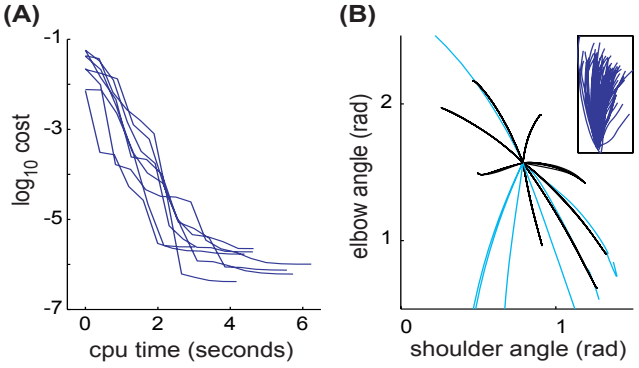

Fig. 5. (A) Cost over iterations of ILQG. (B) Hand paths for random initial conrol laws (inset) and optimized paths (black) to 8 targets. Poor local minima (7 out of 80) are shown in cyan.

Over 90% of the optimization runs (73 out of 80) converged to a solution very similar to the best solution we found for the corresponding target direction. The final trajectories are shown in black in Fig 5B. The remaining 7 runs found local minima (shown in cyan). Therefore, a small number of restarts of ILQG are sufficient to discover what appears to be the global minimum in a relatively complex control problem.

IX. Discussion and extensions

Here we presented a new local method for optimal control of stochastic nonlinear systems subject to control constraints, and applied it to two test problems – a simple pendulum and a high-dimensional biomechanical model of the human arm. In the inverted pendulum problem we demonstrated numerically that the ILQG solutions are close to the global minimum. Additional work is needed to ascertain the properties of the algorithm in more complex problems, where we cannot use global methods for validation. One possibility that we are considering is to take the solution of the ILQG method, apply a stochastic policy gradient algorithm to it (which can be very inefficient, but in the limit of large sample size avoids approximation errors), and see if the ILQG solution is a minimum of gradient descent in the space of feedback control laws. We are also in the process of implementing a very detailed model of the human arm, including 10 degrees of freedom and around 50 muscles; it will be interesting to see if ILQG can handle such a complex system in a reasonable amount of time.

Finally, there are several possible extensions to the work we presented here:

(1) We assumed implicitly that the control system can directly observe the plant state $\mathbf{x}$, while in reality feedback is based on delayed and noisy sensors that may not measure all state variables. It should be possible to extend the algorithm to the partially observable case by combining it with an extended Kalman filter. This will result in a coupled estimation-control problem, which is complicated in the presence of multiplicative noise. However, we have recently solved such a problem for linear systems [16], and are now working on adapting that methodology to the ILQG framework.

(2) As with all second-order methods, convergence is fast near a local minimum but can be slow far away from it. The question then arises, how can we make the algorithm faster at the beginning of the iteration? A very promising idea is to use a multi-resolution method in time: start with a large time step, and gradually reduce it (possibly alternating). With this in mind, we showed explicitly how the LQG approximation depends on the time step Δt – so that the algorithm can be applied without modification with a Δt that varies from one iteration to the next.

(3) While we assumed a specified final time T, the algorithm can be applied in model-predictive mode, using a fixed time horizon rather than a fixed final time. The final cost $h(\mathbf{x})$ will have to be replaced with some approximation to the optimal cost-to-go, but that has to be done whenever fixed-horizon model-predictive control is used.

(4) For deterministic systems ILQG converges to extremal trajectories, just like ODE methods based on Pontryagin's Maximum principle (with the advantage of yielding a feedback controller). In that case the linear-quadratic approximation does not lead to suboptimality of the solution (modulo local minima problems). But for stochastic problems we have no such guarantees, and in fact we know that second-order methods are insensitive to additive noise – suggesting that higher-order approximations to the cost-to-go may be needed. There is a simple way to increase accuracy, by augmenting the state with some nonlinear functions of the state: $\mathbf{z} = [\mathbf{x}; \mathbf{y}]$, where $\mathbf{y} = \mathbf{h}(\mathbf{x})$. Then $\dot{\mathbf{y}} = \partial \mathbf{h}(\mathbf{x}) / \partial \mathbf{x}\, \dot{\mathbf{x}}$, and so the augmented system has well-defined dynamics. In the reaching problem, for example, the cost is a complex trigonometric function in the joint angles but a simple quadratic in the endpoint coordinates. Augmenting the state with those coordinates may therefore increase performance for stochastic problems.

References

[1] D. Bertsekas and J. Tsitsiklis (1996). *Neuro-dynamic programming.* Athena Scientific, Belmont, MA

[2] I. Brown, E. Cheng and G. Loeb (1999). Measured and modeled properties of mammalian skeletal muscle. II. The effects of stimulus frequency on force-length and force-velocity relationships. *J. Muscle Res Cell Motil* 20: 627-643

[3] A. Bryson and Y.-C. Ho (1969). *Applied Optimal Control*. Blaisdell Publishing Company, Massachusetts, USA

[4] C. Harris and D. Wolpert (1998). Signal-dependent Noise Determines Motor Planning. *Nature*, 394: 780-784

[5] D. Jacobson and D. Mayne (1970). *Differential Dynamic Programming*. Elsevier Publishing Company, New York, USA

[6] K. Koerding and D. Wolpert (2004). The loss function of sensorimotor learning. *Proceedings of the National Academy of Sciences* 101: 9839-9842

[7] H. Kushner and P. Dupuis (2001). *Numerical methods for stochastic control problems in continuous time (2nd edition)*. Spinger, New York

[8] W. Li and E. Todorov (2004). Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems. In proceedings of the *1st International Conference on Informatics in Control, Automation and Robotics*, 1: 222-229

[9] L.-Z. Liao and C. Shoemaker (1991). Convergence in unconstrained discrete-time differential dynamic programming. *IEEE Transactions on Automatic Control* 36(6): 692-706

[10] L.-Z. Liao and C. Shoemaker (1993). Advantages of differential dynamic programming over Newton's method for discrete-time optimal control problems. Cornell University Technical Report.

[11] C. Nh, L.-Z. Liao and D. Li (2002). A globally convergent and efficient method for unconstrained discrete-time optimal control. *Journal of Global Optimization* 23: 401-421

[12] J. Pantoja (1988) Differential Dynamic Programming and Newton's method. *Intl J Control* 47: 1539

[13] O. von Stryk (1993). Numerical solution to optimal control problems by direct collocation. *International Series on Numerical Mathematics* 111: 129-143

[14] E. Todorov and M. Jordan (2002). Optimal Feedback Control as a Theory of Motor Coordination. *Nature Neuroscience*, 5(11): 1226-1235

[15] E. Todorov (2004). Optimality principles in sensorimotor control (Review). *Nature Neuroscience* 7(9): 907-915

[16] E. Todorov (2005). Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system. *Neural Computation* 17(5), in press

[17] R. Vinter (2000). *Optimal Control.* Birkhauser, Boston.