Local Optimal Control: Principles and applications

Djordje Mitrovic

February 18, 2010

1 Background

Optimal control theory has received great attention since the 1950s in many fields in science and engineering. In this section we introduce the basics of optimal control theory, whereas more elaborate evaluations of this topic can be found in numerous classic text books [3, 6, 4, 13, 26]. In [29] the authors present a compact up to date overview of optimal control approaches relevant to our work. We focus on the application on biological movement systems, which are covered in depth in [15, 33]. A review of optimality principles with special focus on motor control can be found in [7].

1.1 Optimal control

Let \( x(t) \) denote the state of a plant and \( u(t) \) the applied control signal at time \( t \). In this proposal, we wish to control limb-like systems such as manipulators or arms. The state consists of the joint angles \( q \) and velocities \( \dot{q} \) of a robot, and the actuator control signals \( u \). If the system would be deterministic, we could express its dynamics as \( \dot{x} = f(x,u) \), whereas in the presence of noise we write the dynamics as a stochastic differential equation

\[
dx = f(x,u)dt + F(x,u)d\omega.
\]  

Here, \( d\omega \) is assumed to be Brownian motion noise, which is transformed by a possibly state- and control-dependent matrix \( F(x,u) \). The optimal control problem can be stated as follows: Given an initial state \( x_0 \) at time \( t = 0 \), we seek a control sequence \( u(t) \) such that the system’s state is \( x^* \) at time \( t = T \). Stochastic optimal control theory approaches the problem by first specifying a cost function which is composed of (i) some evaluation \( h(x(T)) \) of the final state, usually penalising deviations from the desired state \( x^* \), and (ii) the accumulated cost \( c(t,x,u) \) of sending a control signal \( u \) at time \( t \) in state \( x \), typically penalising large motor commands. Introducing a policy \( \pi(t,x) \) for selecting \( u(t) \), we can write the expected cost of following that policy from time \( t \) as [34]

\[
\mathbb{E}^\pi(t,x(t)) = \langle h(x(T)) + \int_t^T c(s,x(s),\pi(s,x(s)))ds \rangle.
\]  

1
One then aims to find the policy $\pi$ that minimises the total expected cost $v(\pi(0,x_0))$. Thus, in contrast to classical control, calculation of the trajectory (planning) and the control signal (execution) is not separated anymore, and for example, redundancy can actually be exploited in order to decrease the cost. If the dynamics $f$ is linear in $x$ and $u$, the cost is quadratic, and the noise is Gaussian, the resulting so-called LQG problem is convex and can be solved analytically [26].

However finding an optimal control policy for nonlinear systems is a big challenge [26]. Global solutions could be found in theory by applying dynamic programming methods [3] that are based on the Hamilton-Jacobi-Bellman equations. However, in their basic form these methods rely on a discretisation of the state and action space, an approach that is not viable for large DoF systems. Some research has been carried out on random sampling in a continuous state and action space [27], and it has been suggested that sampling can avoid the curse of dimensionality if the underlying problem is simple enough [1], as is the case if the dynamics and cost functions are very smooth.

1.2 Cost functions in Motor Control

In optimal control problems, a specific cost function needs to be defined as an performance evaluation of the controlled system. In order to achieve smooth point-to-point movements, as observed in humans, [9] presented the minimum jerk model in which the cost function depends on jerk, which corresponds to the squared first derivative of the Cartesian hand acceleration.

$$J = \frac{1}{2} \int_0^T \left( \left( \frac{d^3 x(t)}{dt^3} \right)^2 + \left( \frac{d^3 y(t)}{dt^3} \right)^2 \right) dt \tag{3}$$

Here $x(t), y(t)$ denote the Cartesian coordinates of the hand position at time $t$ and $T$ denotes the final time step. This formulation is independent of the dynamics of the motion system and the optimal trajectory is determined only from the kinematics. Minimum jerk trajectories are straight-lines in task space and follow a bell-shaped velocity profile, properties that match to empirical biological data, recorded in fast reaching movements. However the model fails to explain curved Cartesian trajectories observed in wider movement ranges.

An alternative model, namely the minimum torque-change model, was proposed by [35]. Here the cost function is setup to minimize the rate of change of the torques, and therefore depends on the dynamics of the system rather than only on the kinematics. For a manipulator with $n$-joints the minimum torque-change cost function is defined as

$$J = \frac{1}{2} \int_0^T \sum_{i=1}^n \left( \frac{d^2 \tau_i}{dt^2} \right)^2 dt \tag{4}$$

where $\frac{d^2 \tau_i}{dt^2}$ represents the rate of torque-change in the $i$-th joint. Even though minimum-jerk and minimum torque-change models are capable of predicting many
aspects of biological motion they do not deliver a principled way of how jerk or torque-change could be integrated by the central nervous system (CNS).

The minimum end point variance approach [10] of eye and arm movement incorporates the assumption that the motor control signals are corrupted by noise, the variance of which is proportional to the size of the control signal. Therefore, in the presence of control dependent noise movements that require large motor signals (e.g., fast movements) increase the noise in the systems and lead to deviations of the desired trajectory. In contrast, slow motions would keep the noise level low and lead to more precise motion. Inherently signal-dependent noise can be linked to the speed-accuracy trade-off as described by Fitts’ Law [8]. The cost function relies not on complex parameters, such as torque change and jerk, but rather on the variance of the final position or the consequences of this accuracy. For the CNS this is a more plausible representation of an optimal control problem.

So far no single "correct" cost function could be defined, which explains general biological systems entirely. It is assumed [28] that a general cost function would consist of a mix of cost terms that encode for smoothness, energy consumption and time, and that the cost function may vary during execution. Usually inferring cost functions is done in a propose and evaluate manner, i.e., a cost function is proposed and then evaluated against desired movement data. The inverse approach, this is to infer a cost function from observed data, is much much more difficult and has only been applied on fairly simplistic toy problems [21].

1.3 Optimal Feedback Control

Most optimal motor control models so far have focused on open loop (feed-forward) optimisation. Here, the sequence of motor commands or the trajectory is directly optimised with respect to some cost function, while online feedback is being ignored. Assuming deterministic dynamics (i.e., no unknown perturbations or noise) open-loop control will produce a sequence of optimal motor signals or limb states. However if the system leaves the optimal path it must be corrected for example with a hand tuned PD controller, which most likely will lead to suboptimal behaviour, because the gains are have not been incorporated into the optimisation process. Stable optimal performance can only be achieved by constructing an optimal feedback law that produces a mapping from states to actions by all sensory data available. In these closed loop (feedback) optimisation there is no separation between the trajectory planning and trajectory execution for the completion of a task. In Optimal Feedback Control (OFC), the gains of a feedback controller are optimised to produce an optimal mapping from state to control signals (control law). A key property of OFC is that errors are only corrected by the controller if they adversely affect the task performance, otherwise they are neglected (minimum intervention principle [31]). This is an important property especially in systems that suffer from control dependent noise, since task-irrelevant correction could destabilise the system beside expending additional control effort.

OFC has been used to evaluate various motor behaviours, such as spinal re-
flexes in the limbs of cats during perturbations [11], human postural balance [14] and volitional motor control [18]. In [32] the authors show that an OFC strategy can capture many of the common features of human movement, like goal-directed corrections [17], multi-joint synergies [30] and variable but successful motor performance. A recent experiment supports OFC as a theoretical basis for adaptation in the CNS, by showing that the CNS changes bimanual feedback control optimally according to given task requirements and therefore follows the minimum intervention principle [5]. The OFC framework is currently viewed as the predominant theory for interpreting volitional biological motor control [22], since it unifies motor costs, expected rewards, internal models, noise and sensory feedback into a coherent mathematical framework [23].

While the theoretical importance of OFC to biological motor control is without doubt significant, the available computational methods for solving OFC problems still are not capable of efficiently determining globally valid optimal control laws for large DoF nonlinear systems. In practice the controlled systems often are approximated by linear dynamics in order to make the problem computationally tractable. In [5, 17] the authors successfully used OFC under linear dynamics assumptions to explain human reaching experiments on a high level of observation (i.e., end-effector trajectories, velocity profiles). However if we wish to analyse more detailed effects (e.g., single muscle-signals, co-contraction in joints) for biomechanical systems linearity assumptions in the dynamics may have simplification-artifacts, the effects of which are hard to predict. As we will show later, alternatively iterative optimal control methods can be successfully applied to achieve computationally efficient near optimal behaviour for highly complex systems. However also this method uses local linear approximations to make the OFC problem computationally realisable.

2 iterative Optimal Control Methods

Biologically inspired systems usually have large DoF, typically are highly nonlinear and cannot be represented to fit in the linear quadratic framework. We therefore resort to algorithms that compromise between open loop and closed loop optimisation, that is, algorithms which iteratively compute an optimal trajectory together with a locally valid feedback law. These trajectory-based methods are not directly subject to the curse of dimensionality and still generate locally optimal controllers.

Differential dynamic programming (DDP) [6, 12] is a well-known successive approximation technique for solving nonlinear dynamic optimisation problems. This method uses second order approximations of the system dynamics to perform dynamic programming in the neighbourhood of a nominal trajectory. DDP has second-order convergence and is numerically more efficient than implementations of Newtons method [20]. A more recent algorithm is the iterative Linear Quadratic Regulator (iLQR) [16]. This algorithm uses iterative linearisation of the
nonlinear dynamics around a nominal trajectory, and solves a locally valid LQR problem to iteratively improve the trajectory. However, this method is still deterministic and cannot deal with control constraints or non-quadratic cost functions. A recent extension to iLQR, the iterative Linear Quadratic Gaussian (iLQG) framework [34], allows to model nondeterministic dynamics by incorporating a Gaussian noise model. Furthermore it supports control constraints like non-negative muscle activations or upper control boundaries. The iLQR/iLQG framework showed to be computationally significantly more efficient than DDP [16]. It has also been previously tested on biological motion systems and therefore is the favourite approach for us to investigate further.

2.1 Iterative Linear Quadratic Gaussian - iLQG

This section explains the iLQG framework based on the description given in [34]. We are studying reaching movements of manipulators as a finite time horizon problems of length \( T = k \Delta t \) seconds. Typical values are \( k = 100 \) discretisation steps with simulation rate of \( \Delta t = 0.01 \). For optimising and carrying out a movement, one also has to define a cost function (where also the desired final state is encoded). The expected accumulated cost when following policy \( \pi \) from time \( t \) to \( T \) is

\[
v(\pi)(t, x(t)) = \left< h(x(T)) + \int_t^T L(\tau, x(\tau), \pi(\tau, x(\tau))) d\tau \right>.
\]

(5)

iLQG then finds the control law \( \pi^* \) with minimal \( v(0, x_0) \) by iterating in 4 steps until convergence:

**Step 1:** One starts with an initial time-discretised control sequence \( \bar{u}_k \equiv \bar{u}(k\Delta t) \), which can be chosen arbitrarily (e.g., gravity compensation, or zero sequence). The initial control sequence is applied to the deterministic forward dynamics to retrieve an initial trajectory \( \bar{x}_k \), where

\[
\bar{x}_{k+1} = \bar{x}_k + \Delta t f(\bar{x}_k, \bar{u}_k).
\]

(6)

**Step 2:** By linearising the discretised dynamics (1) around \( \bar{x}_k \) and \( \bar{u}_k \) and by subtracting (6), one gets a dynamics equation for the deviations \( \delta x_k = x_k - \bar{x}_k \) and \( \delta u_k = u_k - \bar{u}_k \):

\[
\delta x_{k+1} = A_k \delta x_k + B_k \delta u_k + C_k(\delta u_k) \xi_k
\]

(7)

\[
A_k = I + \Delta t \frac{\partial f}{\partial x} |_{\bar{x}_k}
\]

(8)

\[
B_k = \Delta t \frac{\partial f}{\partial u} |_{\bar{u}_k}
\]

(9)
The last summand in (7) represents the case when we assume a dynamics model with noise. $\xi^k$ is randomly drawn from a zero mean Gaussian with covariance $\Omega^\xi = I$. $F^{(i)}$ represents the $i$-th column of the matrix $F$.

$$C_k(\partial u_k) = [c_{1,k} + C_{1,k}\partial u_k \ldots c_{p,k} + C_{p,k}\partial u_k]$$

Similarly to the linearised dynamics in (7) one can derive an approximate cost function which is quadratic in $\delta u$ and $\delta x$ such that

$$cost_k = q_k + \delta x^T q_k + \frac{1}{2} \delta x^T Q_k \delta x_k + \delta u^T r_k + \frac{1}{2} \delta u^T R_k \delta u_k + \delta u^T P_k \delta x_k$$

where

$$q_k = \Delta t I; \quad q_k = \Delta t \frac{\partial l}{\partial x}$$

$$Q_k = \Delta t \frac{\partial^2 l}{\partial x \partial x}; \quad P_k = \Delta t \frac{\partial^2 l}{\partial u \partial x}$$

$$r_k = \Delta t \frac{\partial l}{\partial u}; \quad R_k = \Delta t \frac{\partial^2 l}{\partial u \partial u}.$$
a) Compute shortcuts $g, G, H$, by

\[
  g = r_k + B_k^T s_{k+1} + \sum_i C_{i,k}^T S_{k+1} c_{i,k}
\]

(14)

\[
  G = P_k + B_k^T S_{k+1} A_k
\]

\[
  H = R_k + B_k^T S_{k+1} B_k + \sum_i C_{i,k}^T S_{k+1} C_{i,k}
\]

b) Find affine control law by minimising:

\[
  a(\delta u, \delta x) = \delta u^T (g + G \delta x) + \frac{1}{2} \delta u^T H \delta u
\]

(15)

with respect to $\delta u$ leading to

\[
  \delta u = \pi_k(\delta x) = -H^{-1}(g + G \delta x).
\]

(16)

Please note that in equation (16), $H$ is a modified version of $H$ such that there are no negative eigenvalues in $H$ which would make the cost function (arbitrarily) negative. For details about the modification please refer to [34].

c) Update the cost-to-go approximation parameters:

\[
  S_k = Q_k + A_k^T S_{k+1} A_k - G^T H^{-1} G
\]

(17)

\[
  s_k = q_k + A_k^T s_{k+1} - G^T H^{-1} g
\]

\[
  s_k = q_k + s_{k+1} + \sum_i C_{i,k}^T S_{k+1} c_{i,k} - \frac{1}{2} g^T H^{-1} g.
\]

Step 4: After having found the affine control law $\pi(\delta x)$, we apply it to the linearised dynamics (equation (7)) obtaining the optimal control deviations $\delta u(k)$ for each time step $k$ from the nominal sequence $\hat{u}(k)$. We then obtain the new "improved" torque sequence as follows $u_{\text{new}}(k) = \hat{u}(k) + \delta u(k)$. At last we apply $u_{\text{new}}$ to the system dynamics (eq. (1)) and compute the total cost along the trajectory. If the resulting cost has converged (i.e., is not decreasing) iLQG is finished. Otherwise we jump to Step 1 and begin a new iteration with the new control sequence $\hat{u} = u_{\text{new}}$. 

7
Figure 1: Illustration of our iLQG–LD learning and control scheme.

3 Adaptive Optimal Feedback Control

The main theme of this section will focus on incorporating an online model based learning framework within the optimal control paradigm. The motivation for this comes from the current shortcomings of the OFC framework: Firstly, even though OFC is a very attractive control strategy, it has not been studied or tested in practice on large DoF non-linear systems - a step that is essential for enabling wide scale real world application of this attractive control paradigm. Secondly, and more importantly, current implementations assume perfect knowledge of the analytic plant dynamics. This has many disadvantages: The computation of analytical dynamics of complicated, large DoF systems is tedious and in some cases even no forward models are available (e.g., Shadow hand\(^1\)). Moreover, such computations make simplifying rigid body assumptions that are not valid, especially for light-weight flexible joint manipulators. Therefore learning the dynamics using online supervised learning methods are potentially a viable route to account for these issues.

From a biological point of view, enabling OFC to be adaptive would allow us to investigate the role of optimal control in human adaptation scenarios. Indeed, adaptation in humans, for example towards external perturbations, is a key property of human motion and is since nearly two decades a very active area of research [25, 24].

3.1 iLQG–LD

In order to eliminate the need for an analytic dynamics model and to make iLQG adaptive, we wish to learn an approximation \( \tilde{f} \) of the real plant forward dynamics \( \dot{x} = f(x,u) \). Assuming our model \( \tilde{f} \) has been coarsely pre-trained, for example by motor babbling, we can refine that model in an online fashion as shown in Fig. 1. For optimising and carrying out a movement, we have to define a cost function (where also the desired final state is encoded), the start state, and the number of discrete time steps. Given an initial torque sequence \( \bar{u}_k \), the iLQG iterations can be carried out as described in the Section 2.1, but utilising the learned model \( \tilde{f} \). This yields a locally optimal control sequence \( \tilde{u}_k \), a corresponding desired state sequence \( \tilde{x}_k \), and feedback correction gain matrices \( L_k \). Denoting the plant’s true state by \( x \), at each time step \( k \), the feedback controller calculates the required correction to the control signal as \( \delta u_k = L_k(x_k - \tilde{x}_k) \). We then use the final control

\(^1\)www.shadowrobot.com/hand
signal $u_k = \bar{u}_k + \delta u_k$, the plant’s state $x_k$ and its change $dx_k$ to update our internal forward model $\tilde{f}$. As we show in Section ??, we can thus account for (systematic) perturbations and also bootstrap a dynamics model from scratch.

### 3.2 Learning the Dynamics

Various machine learning algorithms can be applied to robot control learning problems. Global learning methods like sigmoid neural networks suffer from the problem of negative interference, i.e., interference between learning in different parts of the input space when input data distributions are not uniform. Local learning methods, in contrast, represent a function by using small simplistic patches - e.g. first order polynomials. The size of the locality is determined by gating activation kernels, and the positions and number of the local kernels are adapted during learning to represent the non-linear function. Because the input data activates only local patches, local learning algorithms are robust against global negative interference. This ensures the flexibility of the learned model towards changes in the dynamics properties of the arm (e.g. load, material wear, and different motion). Furthermore the domain of real-time robot control demands certain properties of a learning algorithm, namely fast learning rates and high prediction speeds at run-time if the model is trained incrementally. Locally Weighted Projection Regression (LWPR) has been shown to exhibit these properties, and to be very efficient for incremental learning of non-linear models in high dimensions [36].

In LWPR, the regression function is constructed by blending local linear models, each of which is endowed with a locality kernel that defines the area of its validity (also termed its receptive field). During training, the parameters of the local models (locality and fit) are updated using incremental Partial Least Squares, and models can be pruned or added on an as-need basis, for example, when training data is generated in previously unexplored regions. Usually the receptive fields of LWPR are modelled by Gaussian kernels, so their activation or response to a query vector $z$ (combined inputs $x$ and $u$ of the forward dynamics $\tilde{f}$) is given by

$$w_k(z) = \exp \left(-\frac{1}{2}(z-c_k)^T D_k(z-c_k)\right), \quad (18)$$

where $c_k$ is the centre of the $k^{th}$ linear model and $D_k$ is its distance metric. Treating each output dimension separately for notational convenience, the regression function can be written as

$$\tilde{f}(z) = \frac{1}{W} \sum_{k=1}^{K} w_k(z) \psi_k(z), \quad W = \sum_{k=1}^{K} w_k(z), \quad (19)$$

$$\psi_k(z) = b_{0k} + b_{1k}^T (z-c_k), \quad (20)$$

where $b_{0k}$ and $b_{1k}$ denote the offset and slope of the $k$-th model, respectively.
LWPR learning has the desirable property that it can be carried out online, and moreover, the learned model can be adapted to changes in the dynamics in real-time. A forgetting factor $\lambda$ [36], which balances the trade-off between preserving what has been learned and quickly adapting to the non-stationarity, can be tuned to the expected rate of external changes. As we will see later, the factor $\lambda$ can be used to model biologically realistic adaptive behaviour to external force-fields.

Despite the large potential learning offers for the OFC framework some critical points should be mentioned. In previous work [19, 36] we have highlighted the curse of dimensionality and consequently, the difficulties involved in learning a perfect model of plant dynamics for a large operating area of a robotic system. This problem is somehow reduced by the fact that high dimensional movement data, as it is produced by humans or humanoid robots, typically is located on a lower dimensional manifold, a property LWPR explicitly exploits. Another significant difference when learning the dynamics for iLQG–LD is it that the priority of the learning task should be set to a wide coverage of the dynamics space rather than solely on a locally very accurate prediction (e.g., for a single trajectory) [2]. Wide receptive fields will produce smooth gradients and allow iLQG–LD to converge "globally". This stands in strong contrast to many classic motor control learning scenarios, in which specific trajectory is learned with high accuracy and as such is required to make the control applicable at all (e.g., learned composite feed-forward component). In practice within iLQG–LD we start off with a crude learned model (i.e., wide kernels without kernel metrics adaptation), which is good enough such that it secures convergence of iLQG–LD. The model then can be refined while using iLQG–LD over a longer period of time.

References


[22] S. H. Scott. Inconvenient truths about neural processing in primary motor


[27] S. Thrun. Monte carlo POMDPs. In S. A. Solla, T. K. Leen, and K. R. Müller,


plex hand manipulation. In *Proceedings of the 26th Annual International
Conference of the IEEE Engineering in Medicine and Biology Society*, pages

[31] E. Todorov and M. Jordan. A minimal intervention principle for coordi-


[33] E Todorov and W Li. Optimal control methods suitable for biomechanical
systems. In *Proceedings of the 25th Annual International Conference of
the IEEE Engineering in Medicine and Biology Society*, pages 1758–1761,
2003.
