Synthesis of complex motor behaviors with optimal control

Emo Todorov

Applied Mathematics, Computer Science & Engineering
University of Washington

Igor Mordatch    Vikash Kumar    Yuval Tassa    Tom Erez    Kendall Lowrey    Paul Kulchenko

NSF, NIH, DARPA
To obtain a concrete theory from the meta-theory of optimal control, we must specify a **cost function** and/or **constraints**. We must also specify a **level of detail** for the (bio)physics model.

As with every scientific theory, we should keep the assumptions as simple and *a priori* justified as possible, and seek predictions that are as elaborate as possible (agreement with data would also be nice).
We represent the movement trajectory in a Fourier basis which guarantees that the constraints are always satisfied. Then we express the cost (integrated squared torque) as a function of the trajectory, and optimize it with a Gauss-Newton method.
How do we know what our theory predicts?

In most fields of science formulating a theory is the hard part, but once it is done, obtaining the corresponding predictions is straightforward.

Optimal control is the opposite: we can easily formulate a cost function, and yet have no idea what the corresponding optimal behavior is.

To obtain predictions from an optimal control model, the scientist has to solve the same optimization problem that the brain is presumably solving... which is challenging because the brain is smarter than the scientist 😊

Solution 1:
Limit ourselves to simple problems, and hope that the “principles” we uncover will miraculously generalize to the real problems the brain is solving.

Solution 2:
Develop algorithms and software that can solve the real problems for us. The mechanisms used by the computer may differ from the brain, but as long as the solutions are similar, we have predictions that can be compared to data.
The cost has the extra term

\[
\sum_t c_{i,\phi(t)}(s) \left( \|e_{i,t}(s)\|^2 + \|\dot{e}_{i,t}(s)\|^2 \right)
\]

where \(e\) is the vector distance to the nearest surface.

The planned contact impulse is penalized in full, but scaled by \(c\) before being applied.

The optimizer has to “declare” the contacts it relies on, forcing it to reason about contact dynamics explicitly.

Mordatch, Todorov and Popovic, SIGGRAPH 2012
Mordatch, Popovic and Todorov, SCA 2012
Extension to musculo-skeletal dynamics

- skeletal dynamics simulated in MuJoCo (our physics engine)
- muscle model based on Wang et al 2012
- metabolic energy model based on Anderson and Pandy 1999
- modifications to the CIO method

\[ f_{pe}(l_{ce}) \]

\[ f_{ce} = f_{lv}(l_{ce}, v_{ce})a \]

\[ f_{se}(l_{se}) \]

Mordatch, Wang, Todorov and Koltun, SIGGRAPH ASIA 2013
Walking at 1.5 m/s

Kinematics

Torques
Running at 4 m/s

Kinematics

Torques

[Graphs showing hip, knee, ankle angles and moments for different muscles during gait cycle]
Our latest algorithms can optimize long and complex movement trajectories, involving rich contact interactions with the environment. This is done fully automatically, given task-level costs and no motion capture, manual scripting or careful initialization.

Presently this optimization runs slower than real-time (~5 min per movement). If it becomes feasible in real-time, most of model-based control will be solved.

How do we get to real-time?
Once every ~30 msec:
- re-optimize the plan up to some horizon (~ 1 sec),
  starting from the current state
- execute the initial portion of the plan,
  while the next plan is being computed

MPC has been used to control chemical plants, play computer chess, drive the Google car. However robot dynamics are too fast for existing optimizers to keep up.

We were able to apply MPC to complex robots for the first time, due to:
  - improved models of contact dynamics;
  - efficient physics simulator (MuJoCo);
  - efficient optimization algorithm (iLQG);
  - selection of cost functions that are realistic yet easier to optimize.
The optimizer evaluates a vast number of candidate control signals in order to find a sequence that works.

This evaluation is done with a physics simulator – which needs to be much faster than real time.

We have developed the first simulator (MuJoCo) that is sufficiently fast and accurate to enable efficient optimization of complex behavior.

Forward dynamics: 100 times faster than real-time.
Inverse dynamics: 1000 times faster than real-time.
A case for MPC in the brain

Complex hardwired behaviors can be generated by “neural machines” with few neurons.

We often think of primate learning as setting up a similar neural machine, which is then responsible for motor execution.

But if that were true, most of the primate brain would shut down during execution of simple familiar movements... and it doesn’t.

Whatever the brain is doing, it appears to be an overkill for executing movements that have already been learned.

What is this processing doing? Unless anyone has a better idea, consider MPC 😊

Most neurons are in the cerebellum – which is presumably a big internal model. This many neurons seem an overkill given the current (limited) views on what internal models are used for.

MPC has a better use for internal models: asking lots of “what-if” questions and thereby optimizing the behavior online. This takes most of the CPU time in our case.
Transfer to the physical world is not easy
\[ \text{belief(state)} = \text{likelihood(sensor data | state)} \times \text{prior(state)} \]

Prior = Physics

Darwin robot:

State estimate:

contact detail
Leave-one-out cross-validation

right knee

left hip pitch

accelerometer x

accelerometer z

gyroscope x

gyroscope z

right heel impulse

left heel impulse

time (s)
Next application: hand manipulation
We use the trajectory optimizer as a **teacher** rather than a demonstrator:
- optimize trajectories with original cost + distance-to-network cost
- train network on resulting trajectories
- iterate
Application to brain-machine interfaces

Controlling a robot with eye movements

Combining speech and eye movements

User + BMI

High-level task specification

Automatic Controller

Computer

Sensors

Robot

Environment