Information Theoretic MPC Using Neural Network Dynamics

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews,
James Rehg, Byron Boots, and Evangelos Theodorou

Georgia Institute of Technology

Introduction

We introduce an information theoretic model predictive control (MPC) algorithm capable of handling complex cost criteria and general nonlinear dynamics. The generality of the approach makes it possible to use multi-layer neural networks as dynamics models, which we incorporate into our MPC algorithm in order to solve model-based reinforcement learning tasks.

Relative Entropy Minimization

In our approach, we minimize the relative entropy between the distribution of trajectories induced by the controller $\mathbb{Q}(\mathbf{u})$ and an optimal distribution $\mathbb{Q}^*$. The form of this distribution can be derived relative to the distribution induced by the uncontrolled dynamics of the system ($\mathbb{P}$):

$$\frac{\mathrm{d}\mathbb{Q}^*}{\mathrm{d}\mathbb{P}} = \frac{\exp\left(-\frac{1}{\lambda}\mathrm{S}(\tau)\right)}{\mathbb{E}_{\mathbb{P}}[\exp\left(-\frac{1}{\lambda}\mathrm{S}(\tau)\right)]} \tag{1}$$

This leads to the following optimization problem:

$$\mathbf{u}^*(\cdot) = \underset{\mathbf{u}(\cdot)}{\operatorname{argmin}}\, \mathbb{D}_{\mathrm{KL}}\left(\mathbb{Q}^* \parallel \mathbb{Q}(\mathbf{u})\right) \tag{2}$$

whose solution is a path integral, which can be approximated by sampling trajectories from the system dynamics and taking a cost-weighted average.

Learning Dynamics with Neural Networks

In order to sample system trajectories, we need to learn a system model. Given that the state $\mathbf{x}$ is partitioned as $\mathbf{x} = (\mathbf{q}, \dot{\mathbf{q}})$, where $\mathbf{q}$ is the configuration of the system and $\dot{\mathbf{q}}$ is its time derivative, we seek a function $\mathbf{f}$ so that the full state transition is:

$$\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t, \mathbf{u}_t) = \begin{bmatrix} \mathbf{q}_t + \dot{\mathbf{q}}_t \Delta t \\ \dot{\mathbf{q}}_t + \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t)\Delta t \end{bmatrix}$$

where Δt is a discrete time increment. We represent $\mathbf{f}$ with a neural network and train it on a dataset of state-action-acceleration triplets $\mathcal{D} = \{(\mathbf{x}_t, \mathbf{u}_t, (\dot{\mathbf{q}}_{t+1} - \dot{\mathbf{q}}_t)/\Delta t)\}_t$.

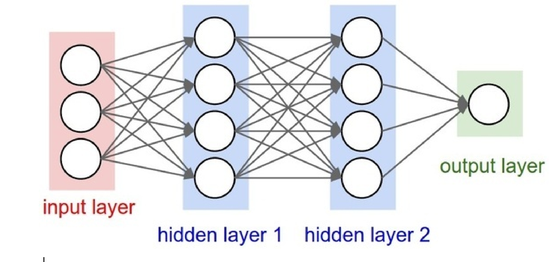

Figure 1 : Top: Fully connected multi-layer neural network architecture. Bottom: Actual vs. predicted sequence with a neural network dynamics model.

MPPI Algorithm

We place our optimization scheme with learned neural network dynamics in a model predictive control setting. In this setting, optimization takes place on the fly: thousands of trajectories are sampled each time-step (20 ms) in order to compute an update to the control sequence, then a single control input is executed before re-optimization occurs.

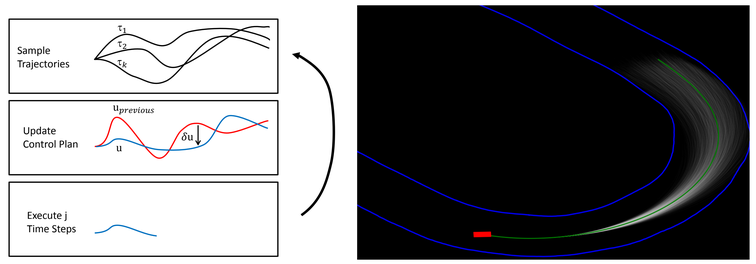

Figure 2 : Left: A single MPPI iteration. Right: Spray of sampled trajectories for the AutoRally system.

The bulk of the computation involved in this model predictive path integral control (MPPI) algorithm can be done in parallel, which we do using a GPU.

Experiments

We tested the MPPI algorithm with a neural network model using simulated cart-pole swing-up and quadroter navigation tasks, as well as on real robotics hardware in an aggressive driving task with the Georgia Tech AutoRally platform. We used fully connected networks with two (equal sized) hidden layers and tanh non-linearities.

Figure 3 : AutoRally platform power sliding around a corner with the MPPI controller.

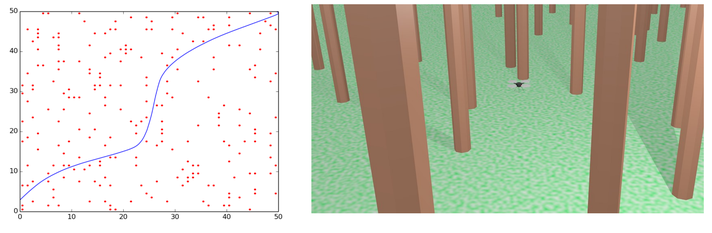

Figure 4 : Left: Top down view of obstacle field for quadrotor navigation task. Right: Visualization of the quadrotor simulation.

Results

In the simulation tasks, the controller with the neural network model comes within 10% of the performance with the ground truth model. This is good enough to swing up (and stabilize) the cart-pole and to navigate the quadrotor through the obstacle field. The controller with the neural network model on the AutoRally platform consistently achieves speeds over 8 m/s, and intelligently modulates its speed around corners.

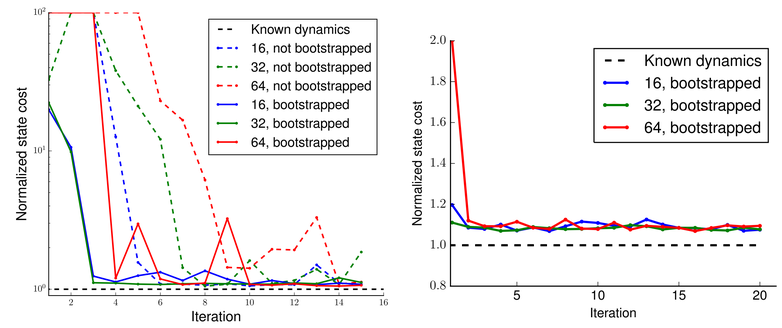

Figure 5 : Relative cost for MPPI with a neural net model vs. a ground truth model for the cart-pole (left) and quadcopter (right)

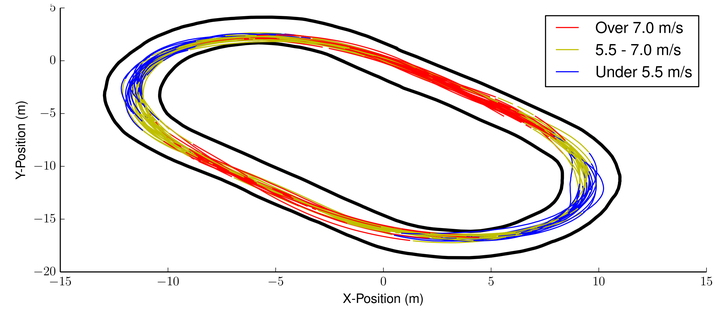

Figure 6 : Top Left: Sample quadropter trajectory through an obstacle field. Top Right: Quadrotor simulation environment. Bottom: Speed profiles for the AutoRally.

Table 1 : Training and test run statistics on the AutoRally platform.

Performance Metric	9 m/s target	11 m/s target
Top Speed	8.13 m/s	8.71 m/s
Best Lap Time	10.32s	9.43s
Maximum Side-Slip Angle	22.14°	34.65°

Big Picture

Given an approximate model in the form of a neural network, we can perform model predictive control for a variety of complex systems and tasks. All of the behaviors are generated on the fly and are close to optimal with respect to the learned dynamics.

References

[1] Williams G, Wagener N, Goldfain B, Drews P, Rehg J.M, Boots B, and Theodorou E.A "Information Theoretic MPC for Model Based Reinforcement Learning". IEEE International Conference on Robotics and Automation, Singapore, 2017.

[2] Williams G, Drews P, Goldfain B, Rehg J.M, and Theodorou E.A "Aggressive Driving with Model Predictive Path Integral Control". IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 2016.