Curious iLQR: Resolving Uncertainty in Model-based RL

Sarah Bechtle[1,2]**, Yixin Lin**[2]**, Akshara Rai**[2]**, Ludovic Righetti**[1,3] **and Franziska Meier**[2]

[1]Max Planck Institute for Intelligent Systems
[2]Facebook AI Research, [3]New York University
{sbechtle,ludovic.righetti}@tuebingen.mpg.de
{yixinlin,akshararai,fmeier}@fb.com

Abstract: Curiosity as a means to explore during reinforcement learning problems has recently become very popular. However, very little progress has been made in utilizing curiosity for learning control. In this work, we propose a model-based reinforcement learning (MBRL) framework that combines Bayesian modeling of the system dynamics with *curious iLQR*, an iterative LQR approach that considers model uncertainty. During trajectory optimization the *curious iLQR* attempts to minimize both the task-dependent cost and the uncertainty in the dynamics model. We demonstrate the approach on reaching tasks with 7-DoF manipulators in simulation and on a real robot. Our experiments show that MBRL with *curious iLQR* reaches desired end-effector targets more reliably and with less system rollouts when learning a new task from scratch, and that the learned model generalizes better to new reaching tasks.

Keywords: Exploration, Robots, Model-based RL

1 Introduction

Model-based reinforcement learning holds promise for sample-efficient learning on real robots [1]. The hope is that a model learned on a set of tasks can be used to learn to achieve new tasks faster. A challenge is then to ensure that the learned model generalizes beyond the specific tasks used to learn it. We believe that curiosity, as means of exploration, can help with this challenge. Though curiosity has been defined in various ways, it is generally considered a fundamental building block of human behaviour [2] and essential for the development of autonomous behaviour [3].

In this work, we take inspiration from [4], which defines curiosity as motivation to resolve uncertainty in the environment. Following this definition, we postulate that by seeking out uncertainties, a robot is able to learn a model faster and therefore achieve lower costs more quickly compared to a non-curious robot. Keeping real robot experiments in mind, our goal is to develop a model-based reinforcement learning (MBRL) algorithm that optimizes action sequences to not only minimize a task cost but also to reduce model uncertainty.

Specifically, our MBRL algorithm iterates between learning a probabilistic model of the robot dynamics and using that model to optimize local control policies (i.e. desired joint trajectories and feedback gains) via a *curious* version of the iterative Linear Quadratic Regulator (iLQR) [5]. These policies are executed on the robot to gather new data to improve the dynamics model, closing the loop, as summarized in Figure 1.

In a nutshell, our curious iLQR aims at optimizing local policies that minimize the cost *and* explore parts of the model with high uncertainty. In order to encourage actions that explore states for
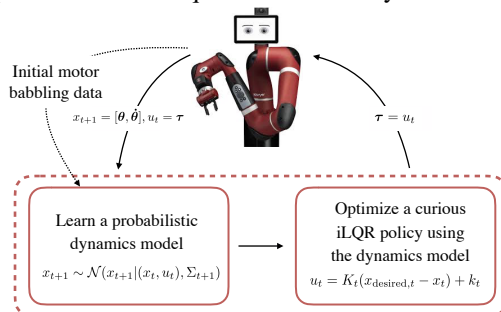

Figure 1: Approach overview: motor babbling data initializes the dynamic model, the main loop then alternates between model learning and policy updates.

which the dynamics model is uncertain, we incorporate the variance of the model predictions into the cost function evaluation. We propose a computationally efficient approach to incorporate this uncertainty by leveraging results on risk-sensitive optimal control [6, 7]. [6] showed that optimizing actions with respect to the expected exponentiated cost directly takes into account higher order moments of the cost distribution while affording the explicit computation of the optimal control through Riccati equations. A risk-sensitive version of iLQR was recently proposed in [7]. While in these approaches the dynamic model is typically considered known and uncertainty comes from external disturbances, we propose to instead explicitly incorporate model uncertainty in the algorithm to favor the exploration of uncertain parts of the model. The proposed coupling between model learning and risk-sensitive control explicitly favours actions that resolve the uncertainty in the model while minimizing a task-related cost.

The contributions of this work are as follows: 1) We present a MBRL algorithm that learns a global probabilistic model of the dynamics of the robot from data and show how to utilize the uncertainty of the model for exploration through our curious iLQR optimization. 2) We demonstrate that our MBRL algorithm can scale to seven degree of freedom (DoF) manipulation platform in the real world without requiring demonstrations to initialize the MBRL loop. 3) The results show that using curiosity not only learns a better model faster on the initial task, but also that this model generalizes to new tasks more reliably. We perform an extensive evaluation in both simulation and on hardware.

2 Background

The goal of MBRL is to solve a task through learning a model f of the true dynamics f_{real} of the system that is subsequently used to solve an optimal control problem. The dynamics are described through $x_{t+1} = f(x_t, u_t)$ where x_t and u_t are the state and action of the current time step, and x_{t+1} the state at the next time step. f represents the learned model of the dynamics. MBRL seeks to find a policy $u_t = \pi(x_t)$ that minimizes a cost $\mathcal{J}(x_t, u_t)$ describing the desired behavior. Policy optimization can be performed in various ways such as trajectory sampling approaches as summarized and evaluated in [8], random shooting methods, where trajectories are randomly chosen and evaluated with the learned model, or iterative LQG approaches, as in [9]. Model learning also can be tackled with various methods. [10] proposes learning linear models of the forward dynamics. In [8] the dynamics are learned with an ensemble of neural networks. In general, the learned model of dynamics can be deterministic as in [10] or probabilistic as in [11, 8].

In MBRL, the learned model is used to simulate the robot behaviour when optimizing a trajectory or control policy. The learned model and the optimizer are task independent; this independence promises sample efficiency and generalization capabilities, as an already learned model can be reused for new tasks. As a side effect, however, the learned models quality can drastically affect the computed solution, as pointed out in [12, 1, 13], since the policy is optimized given the current learned model and not by interacting with the robot. This effect is called model bias [13] and can lead to a policy with drastically lower performance on the real robot. We argue that exploration can alleviate this model-bias. Resolving model uncertainty while optimizing for a task can encourage visiting states which resolve ambiguities in the learned model and therefore lead to both better models and control policies.

2.1 Intrinsic motivation for RL

The concept of curiosity has also been explored within the reinforcement learning literature from various angles. For example, a first attempt towards intrinsically motivated agents consisted in rewarding agents to minimize prediction errors of sensory events [14, 15, 16]. This initial work was designed for low-dimensional and discrete state-and-action spaces. Recently, curiosity as a means to better explore was also investigated for high-dimensional continuous state spaces [17, 18]. Most of this work, including recent efforts towards curiosity driven robot learning [19, 20], has defined curiosity as a function of model prediction error and within a model-free reinforcement learning framework. In MBRL, [21] recently proposed a measure of disagreement as exploration signal. [22] propose a maximum entropy exploration behaviour. Other algorithms which take uncertainty into account have been presented as well [11, 23, 8, 24]. They differ in their choice of policy optimization, dynamics model representation and how they incorporate uncertainty. While [11, 8] utilize model uncertainty to generate trajectory distributions, the uncertainty does not play an explicit

role in the cost. Thus, these approaches do not explicitly optimize actions that resolve uncertainty in the current model of the dynamics, which is in contrast to the approach we propose in this paper.

2.2 Risk Sensitive stochastic optimal control

Risk-sensitive optimal control has a long history [6, 25]. The central idea is to not only minimize the expectation of the performance objective under the stochastic dynamics but to also take into account higher-order moments of the cost distribution. The objective function takes the form of an exponential transformation of the performance criteria $J = \min_\pi \mathbb{E}\{\exp[\sigma \mathcal{J}(\pi)]\}$ [6]. Here, $\mathcal{J}(\pi)$ is the performance index, which is a random variable, and a functional of the policy π. $\mathbb{E}$ is the expected value of $\mathcal{J}$ over stochastic trajectories induced by the policy π. $\sigma \in \mathbb{R}$ accounts for the sensitivity of the cost to higher order moments (variance, skewness, etc.). Notably, from [7], the cost is $\frac{1}{\sigma}\log(J) = \mathbb{E}(\mathcal{J}^*) + \frac{\sigma}{2}\text{var}(\mathcal{J}^*) + \frac{\sigma^2}{6}\text{sk}(\mathcal{J}^*) + \cdots$, where var and sk stand for variance and skewness and $\mathcal{J}^*$ is the optimal task cost. When $\sigma > 0$ the optimal control will be risk-averse, favoring low costs with low variance but when $\sigma < 0$ the optimal control will be risk-seeking, favoring low costs with high variance. $\sigma = 0$, reduces to the standard, risk-neutral, optimal control problem. Jacobson [6] originally demonstrated that for linear dynamics and quadratic costs the optimal control could be computed as the solution of a Riccati equation. Leveraging this result, [7] recently proposed a risk-sensitive extension of iLQR and [26] further extended the approach to explicitly incorporate measurement noise.

3 MBRL via Curious iLQR

We present our approach to incorporate curious behaviour into a robot's learning control loop. We are interested in minimizing a performance objective $\mathcal{J}$ to achieve a desired robot behavior and approximate the true dynamics of the system with a discrete-time dynamical system

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

where $\mathbf{x}_t$ denotes the state of the system at time step t and $\mathbf{f}$ represents the unknown model of the dynamics of the system and needs to be learned to achieve the desired task. The hypothesis we seek to confirm is that, by trying to explore uncertain parts of the model, our MBRL algorithm can learn a good dynamics model more quickly and find behaviors with higher performance. Our algorithm learns a probabilistic model of the system dynamics while concurrently optimizing a desired cost objective (Figure 1). It combines i) a risk-seeking iLQR algorithm and ii) a probabilistic model of the dynamics. We describe the algorithm in the following. In particular, we show how to incorporate model uncertainty in risk-sensitive optimal control. Algorithm 1 shows the complete algorithm.

Algorithm 1 MBRL Algorithm

1: $\mathcal{D} \leftarrow$ motor babbling data
2: train model f on D
3: **while** $i <$ iter **do**
4:     $\pi \leftarrow$ optimize policy via Alg 3
5:     $D_{\text{new}} \leftarrow$ rollout π on system
6:     $\mathcal{D} = \mathcal{D} \cup D_{\text{new}}$
7:     train model f on $\mathcal{D}$
8: **end while**

Algorithm 2 simulate-policy(x, τ, k, K, α)

1: $x_0^{\text{new}} \leftarrow x_0$
2: **while** $t < T$ **do**
3:     $\tau_t^{\text{new}} \leftarrow \tau_t + \alpha k_t + K_t(x_t - x_t^{\text{new}})$
4:     $x_{t+1}^{\text{new}} \leftarrow f(x_t^{\text{new}}, \tau_t)$
5: **end while**
6: return $\tau^{\text{new}}, x^{\text{new}}$

Algorithm 3 curious-iLQR

1: $\tau \leftarrow$ Initial random torque trajectory
2: $x^* \leftarrow$ unroll τ using f
3: $\mathbb{A} \leftarrow$ Line search parameters, $[0,\ldots,1]$
4: $J^* \leftarrow$ Optimal iLQR cost so far
5: **while** $i <$ opt iter **do**
6:     $k, K \leftarrow$ backward pass, see 3.2
7:     **for** $\alpha \in \mathbb{A}$ **do**
8:         $\tau^{new}, x^{new} \leftarrow$ simulate-policy(x, τ, k, K, α)
9:         $J_{new} \leftarrow$ Compute cost of $\tau^{\text{new}}, x^{\text{new}}$
10:         **if** $J_{new} < J^*$ **then**
11:             $\tau, x^* \leftarrow x^{\text{new}}, \tau^{new}$
12:         **end if**
13:         **if** converged **then**
14:             return $\pi(x) = \tau + K(x - x^*)$
15:         **end if**
16:     **end for**
17: **end while**

3.1 Risk-sensitive iLQR

Consider the following general nonlinear stochastic difference equation

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t)\Delta t + \mathbf{g}(\mathbf{x}_t, \mathbf{u}_t)\Delta\omega \tag{2}$$

where $\mathbf{g}$ maps a Brownian motion $\Delta\omega$, with 0 mean and covariance $(\mathbf{\Sigma} \cdot \Delta t)$, to system states. $\Delta\omega$ and the nonlinear map $\mathbf{g}$, typically model an unknown physical disturbance, while assuming a known model $\mathbf{f}$ of the dynamics. When considering the exponentiated performance criteria $J = \min_\pi \mathbb{E}\{\exp[\sigma\mathcal{J}(\pi)]\}$ (see 2 for more details), it has been shown that iLQR [5] can be extended to risk-sensitive stochastic nonlinear optimal control problems [7]. The algorithm begins with a nominal state and control input trajectory $\mathbf{x^n}$ and $\mathbf{u^n}$. The dynamics and cost are approximated to first and second order respectively along the nominal trajectories $\mathbf{u_t^n}, \mathbf{x_t^n}$ in terms of state and control deviations $\delta\mathbf{x_t} = \mathbf{x_t} - \mathbf{x_t^n}, \delta\mathbf{u_t} = \mathbf{u_t} - \mathbf{u_t^n}$. Given a quadratic control cost, the locally optimal control law will be of the form $\delta\mathbf{u_t} = \mathbf{k_t} + \mathbf{K_t}\delta\mathbf{x_t}$. The underlying optimal control problem can be solved by using Bellman equation

$$\Psi_\sigma(\delta\mathbf{x_t}, \mathbf{t}) = \min_{\mathbf{u}}\{l(\mathbf{x}, \mathbf{u}, \mathbf{t}) + \mathbb{E}[\Psi_\sigma(\delta\mathbf{x_{t+1}}, \mathbf{t+1})]\} \tag{3}$$

where l is the quadratic cost, and by making the following quadratic approximation of the value function $\Psi(\delta\mathbf{x_t}, \mathbf{t}) = \frac{1}{2}\delta\mathbf{x_t^T}\mathbf{S_t}\delta\mathbf{x} + \delta\mathbf{x_t^T}\mathbf{s_t} + s_t$ where $\mathbf{S_t} = \nabla_{\delta x \delta x}\Psi$ and $\mathbf{s_t} = \nabla_{\delta x}\Psi - \mathbf{S_t}\delta\mathbf{x_t}$ are functions of the partial derivatives of the value function.

Using the (time-varying) linear dynamics, the quadratic cost and the quadratic approximation of Ψ, and solving for the optimal control, we get

$$\delta\mathbf{u_t} = \mathbf{k_t} + \mathbf{K_t}\delta\mathbf{x_t}, \quad \mathbf{k_t} = -\mathbf{H_t^{-1}}\mathbf{g_t}, \quad \text{and} \quad \mathbf{K_t} = -\mathbf{H_t^{-1}}\mathbf{G_t} \tag{4}$$

where $\mathbf{H_t}, \mathbf{g_t}, \mathbf{G_t}$ are given by

$$\mathbf{H_t} = \mathbf{R_t} + \mathbf{B_t^T}\mathbf{S_t}\mathbf{B_t} + \sigma\mathbf{B_t^T}\mathbf{S_t^T}\mathbf{C}\mathbf{\Sigma_{t+1}}\mathbf{C^T}\mathbf{S_t}\mathbf{B_t}$$
$$\mathbf{g_t} = \mathbf{r_t} + \mathbf{B_t^T}\mathbf{s_t} + \sigma\mathbf{B_t^T}\mathbf{S_t^T}\mathbf{C}\mathbf{\Sigma_{t+1}}\mathbf{C^T}\mathbf{s_t} \tag{5}$$
$$\mathbf{G_t} = \mathbf{P_t^T} + \mathbf{B_t^T}\mathbf{S_t}\mathbf{A_t} + \sigma\mathbf{B_t^T}\mathbf{S_t^T}\mathbf{C}\mathbf{\Sigma_{t+1}}\mathbf{C^T}\mathbf{S_t}\mathbf{A_t}$$

where $\mathbf{A_t} = \Delta t\frac{\partial\mathbf{f}}{\partial\mathbf{x_t}}, \mathbf{B_t} = \Delta t\frac{\partial\mathbf{f}}{\partial\mathbf{u_t}}$ and $\mathbf{q_t}, \mathbf{r_t}, \mathbf{Q_t}, \mathbf{R_t}$ and $\mathbf{P_t}$ are the coefficients of the Taylor expansion of the cost function around the nominal trajectory. The corresponding backward recursions are

$$\mathbf{s_t} = \mathbf{q_t} + \mathbf{A_t^T}\mathbf{s_{t+1}} + \mathbf{G_t^T}\mathbf{k_t} + \mathbf{K_t^T}\mathbf{H_t}\mathbf{k_t} + \sigma\mathbf{A_t^T}\mathbf{S_{t+1}^T}\mathbf{C}\mathbf{\Sigma_{t+1}}\mathbf{C^T}\mathbf{s_{t+1}} \tag{6}$$

$$\mathbf{S_t} = \mathbf{Q_t} + \mathbf{A_t^T}\mathbf{S_{t+1}}\mathbf{A_t} + \mathbf{K_t^T}\mathbf{H_t}\mathbf{K_t} + \mathbf{G_t^T}\mathbf{K_t} + \mathbf{K_t^T}\mathbf{G_t} + \sigma\mathbf{A_t^T}\mathbf{S_{t+1}^T}\mathbf{C}\mathbf{\Sigma_{t+1}}\mathbf{C^T}\mathbf{S_{t+1}}\mathbf{A_t} \tag{7}$$

We note that this Riccati recursion is different from usual iLQR ([5]) due to the presence of the covariance Σ: the locally optimal control law explicitly depends on the noise uncertainty.

3.2 Curious iLQR: seeking out uncertainties

We use Gaussian Process (GP) regression to learn a probabilistic model of the dynamics in order to include the predictive variance from the model into the risk-sensitive iLQR algorithm. This predictive variance will then capture both model as well as measurement uncertainty. Specifically, we set $\mathbf{x_t} = [\boldsymbol{\theta_t}, \dot{\boldsymbol{\theta}}_\mathbf{t}]$ where $\boldsymbol{\theta_t}, \dot{\boldsymbol{\theta}}_\mathbf{t}$ are joint position and velocity vectors respectively. We let $\mathbf{u_t}$ denote the vector of commanded torques. After each system rollout, we get a new set of tuples of states and actions $(\mathbf{x_t}, \mathbf{u_t})$ as inputs and $\ddot{\boldsymbol{\theta}}_\mathbf{t+1}$, joint accelerations at the next time step, as outputs which we add to our dataset $\mathcal{D}$ on which we re-train the probabilistic dynamics model (see Algorithm 1). Once trained, the model produces a one step prediction of the joint accelerations of the robot as a probability distribution of the form

$$p(\ddot{\boldsymbol{\theta}}_\mathbf{t+1}|\mathbf{x_t}, \mathbf{u_t}) = \mathcal{N}(\ddot{\boldsymbol{\theta}}_\mathbf{t+1}|\mathbf{h}(\mathbf{x_t}, \mathbf{u_t})\Delta t, \mathbf{\Sigma_{t+1}}) \tag{8}$$

where $\mathbf{h}$ is the mean vector and $\mathbf{\Sigma_{t+1}}$ the covariance matrix of the predictive distribution evaluated at $(\mathbf{x_t}, \mathbf{u_t})$. The outputs is the acceleration at the next time step $\ddot{\boldsymbol{\theta}}_\mathbf{t+1}$ which is numerically integrated to velocity $\ddot{\boldsymbol{\theta}}_\mathbf{t+1}\Delta t + \dot{\boldsymbol{\theta}}_\mathbf{t} = \dot{\boldsymbol{\theta}}_\mathbf{t+1}$ and position $\dot{\boldsymbol{\theta}}_\mathbf{t+1}\Delta t + \boldsymbol{\theta_t} = \boldsymbol{\theta_{t+1}}$. This results in a Gaussian predictive distribution of the system dynamics $\mathbf{f}$

$$\mathbf{x_{t+1}} \sim \mathcal{N}(\mathbf{x_{t+1}}|\mathbf{x_t} + \mathbf{h}(\mathbf{x_t}, \mathbf{u_t})\Delta t, \mathbf{\Sigma_{t+1}}) \tag{9}$$

It is the covariance matrix Σ_{t+1} of this distribution that is incorporated into the Riccati equations from above. Specifically, during each MBRL iteration we optimize a new local feedback policy

under the current dynamics model $\mathbf{f}$, via Algorithm 3. Each outer loop of the optimization, re-linearizes $\mathbf{f}$ with respect to the current nominal trajectories $\mathbf{u_t^n}$, $\mathbf{x_t^n}$ in the backward-pass:

$$\delta\mathbf{x_{t+1}} = \mathbf{A_t}\delta\mathbf{x_t} + \mathbf{B_t}\delta\mathbf{u_t} + \mathbf{C_t}\omega_t \tag{10}$$

with $\mathbf{A_t} = \Delta t \frac{\partial \mathbf{f}}{\partial \mathbf{x_t^n}}$, $\mathbf{B_t} = \Delta t \frac{\partial \mathbf{f}}{\partial \mathbf{u_t^n}}$ and $\omega_t \sim \mathcal{N}(\omega_t|0, \mathbf{\Sigma_{t+1}})$, where $\mathbf{A_t}$ and $\mathbf{B_t}$ are the analytical gradients of the probabilistic model prediction at each time step and $\mathbf{C_t}$ weights how the uncertainty is propagated trough the system. We utilize the Riccati equations from Section 3.1, Equations (5) and (6), to optimize a new local feedback policy that utilizes the models predictive covariance Σ_{t+1}. During the shooting phase of the algorithm, we integrate the nonlinear model from the GP and, to guarantee convergence to lower costs, we use a line search approach during the optimization. We leverage the risk-seeking capabilities of the optimization by setting $\sigma < 0$. The algorithm then favors costs with higher variance which is related to exploring regions of the state space with higher uncertainty in the dynamics. As a result, the agent is encouraged to select actions that explore uncertain regions of the dynamic model while still trying to reduce the task specific error. With $\sigma = 0$ the agent will ignore any uncertainty in the environment and therefore not explore. This is equivalent to standard iLQR optimization which ignores higher order statistics of the cost function. An overview of *curious iLQR* is given in Algorithm 3.

4   Illustration: Curious iLQR

In this section, we want to illustrate the advantages of using the motivation to resolve model uncertainty as an exploration tool. The objectives of this section is to give an intuitive example of the effect of our MBRL loop. In the following, and throughout the paper, we will refer to the agent that tries to resolve the uncertainty in its environment as curious and the one that is not following the uncertainty but only optimizes for the task related cost as normal.
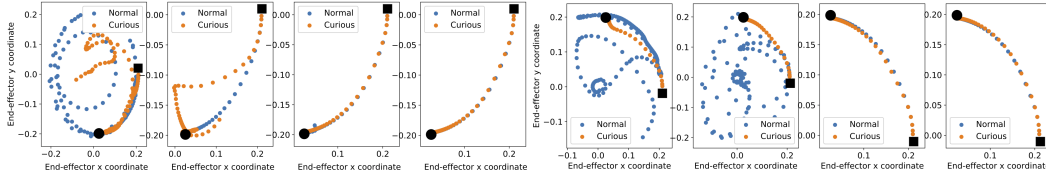

Figure 2: End-effector position of curious and normal agent for 4 learning iterations on 2 different targets. The targets are represented by the black dots, the starting position by the black squares.

The experimental platform is the OpenAI Gym Reacher environment [27], a two degrees of freedom arm attached at the center of the scene. The goal of the task is to reach a target placed in the environment. In the experiments presented here, actions were optimized as described in section 3. The probabilistic model was learned with Gaussian Process (GP) regression using the GPy library [28]. The intuition behind this experiment is that, if an agent is driven to resolve uncertainty in its model, a better model of the system dynamics can be learned and therefore used to optimize a control sequence more reliably. Our hypoth-

Figure 3: Reacher performance /10 trials.

esis is that, the model learned by the curious agent is better by the end of learning and therefore we expect it to perform better when using it to solve new reaching tasks. In Figure 2 we show the result-
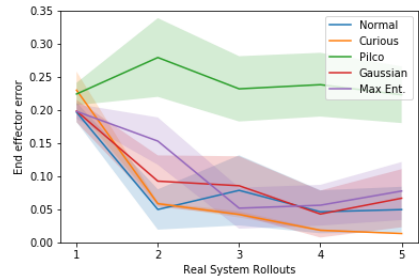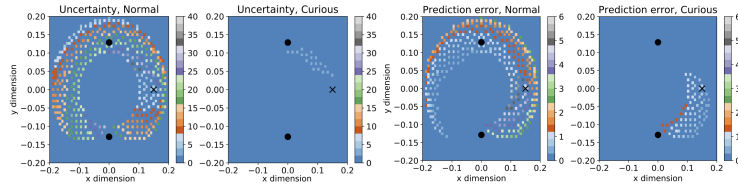

Figure 4: The uncertainty and the prediction error in end-effector space after training, for the normal and curious agent. The cross is the initial position. Regions that are not by the arm reachable are shown in blue.

ing end-effector trajectories of 8 consecutive MBRL iterations when optimizing to reach 2 different targets in sequence. We compare the behavior of the curious and normal agent in orange and blue,

respectively. The targets are represented by the black dot. The curious agent tries to resolve the uncertainty within the model; the normal agent optimizes only for the task related cost. The normal agent seemingly reaches the first target after the second learning iteration; the curious agent only manages to reach the target during the third iteration. Interestingly, the exploration of the curious agent leverages the arm to reach the second target immediately and continues to reach it consistently thereafter. Figure 4 confirms the intuition that the curious agent has learned a better model than the normal agent. The figure shows the uncertainty and the prediction error (in end-effector space) of the model learned by the normal and the curious agent respectively. With curiosity, the learned model has overall lower uncertainty and prediction error values over the whole state space. We also compare our MBRL loop via *curious iLQR* optimization to: normal iLQR, a random exploration controller that adds Gaussian noise to the actions with mean 0 and variance 0.2, a maximum entropy exploration behaviour following the approach proposed in [22] and PILCO [11], in Figure 3. For these experiments, we initialize the model with only two data points collected randomly during motor babbling. We report the mean and the standard deviation across 10 trials, where each trial starts from a different initial joint configuration and is initialized with a different initial torque trajectory for optimization. In this scenario, with a very poor initial model quality, PILCO could not perform comparably to our MBRL loop. MBRL via *curious iLQR* outperforms all the other approaches. Furthermore it converges to solutions more reliably, as the variance between trials is lowest.

5 Experiments on high-dimensional problems

Finally, the goal of this work is to learn motor skills on a torque-controlled manipulator. Our experimental platform is the Sawyer robot from Rethink Robotics [29], a 7 degrees of freedom manipulator. We start with experiments performed in the PyBullet physics simulator [30]. In the next Section, we present results on the Sawyer robot arm hardware. Previous work such as [7] and [26], which use risk-sensitive control variations of iLQR, primarily deal with simplified, low dimensional problems. Our experiments are conducted on a 7 degree of freedom robot, and the higher dimensional system adds some complexities to the approach: the gradients in Section 3.1 of the value function (Equations (6), (7)) tend to suffer from numerical ill-conditioning in high-dimensions. We account for this issue with Tikhonov regularization: before inversion for calculating the optimal control we add a diagonal matrix to $\mathbf{H_k}$ from Equation (5). The regularization parameter and the line search parameter α are adapted following the Levenberg Marquardt heuristic [5].

The goal of these experiments is to reach a desired target joint configuration θ. We show results for dynamics learned with GP regression (GPR), as well as initial results on ensemble of probabilistic neural networks (EPNN) following the approach presented in [31]. When using GPs, a separate GP is trained for each output dimension.

We perform two sets of experiments, both in simulation and on hardware, to analyze the effect of using curiosity. Specifically, we believe that curiosity helps to find better solutions faster, because it guides exploration within the MBRL loop. Intuitively, curiosity helps to observe more diverse data samples during each rollout such that the model learns more about the dynamics.

We start with evaluation in simulation. Throughout all of the simulation experiments the optimization horizon was 150 time steps long at a sampling rate of 240 Hz. Motor babbling was performed at the beginning for $0.5s$ by commanding random torques in each joint.

5.1 Reaching task from scratch

During the first set of experiments, we compare the performance when learning to reach a given target configuration from scratch. We compare our MBRL loop, as before, when using our *curious iLQR* optimization, regular iLQR, a random exploration controller and a maximum entropy exploration behaviour as described previously. PILCO was not able to learn the reaching movement on the 7-DoF manipulator, so we exclude the results from the analysis. We perform this experiments for each kind of controller 5 times. Each run slightly perturbs the initial joint positions, and uses a random initial torque trajectories for the optimizer. For a given target joint configuration, 5 iterations of optimizing the trajectory, running the trajectory on the system and updating the dynamics model, were performed. We perform this experiment for 3 different target joint configurations. The following results are averaged across the $5x3$ runs (5 runs per target). The left most plot in Figure 5 compares performance of *curious iLQR* when using EPNN vs GPR for dynamics model learning,
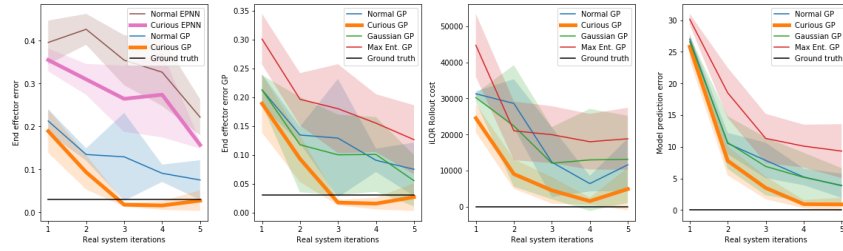
Figure 5: Distance in end effector space for EPNN vs. GP in m (1). Distance in end effector space in m (2), iLQR rollout cost (3) and model prediction error (4) with the GP model, compared to our baselines.

with and without curiosity. Our analysis shows that MBRL via *curious iLQR* improves performance over regular iLQR, for both model architectures. While the EPNN is more promising in scaling our approach, it currently requires more data to train. For this reason we will focus on the GP model for the remainder of our experimental section. In the 2nd to 4th plot of Figure 5, we compare the performance of *curious iLQR* against the above mentioned baselines for exploration during policy optimization, when using GPR for model learning. We compare the methods with respect to 3 metrics: final Euclidean end-effector distance (plot 2), iLQR cost (plot 3) and the predictive performance of the model on each rollout (plot 4). We can consistently see that, on average, MBRL via *curious iLQR* outperforms the other approaches: the error/cost is smaller and the solutions are more consistent across trials as the standard deviation is lower. This shows that curiosity can lead to faster learning of a new task, when learning from scratch. The results on the predictive performance of the model suggest that the quality of the model learned via *curious iLQR* might be better in terms of generalization. In the next section we present results that investigate this assumption.

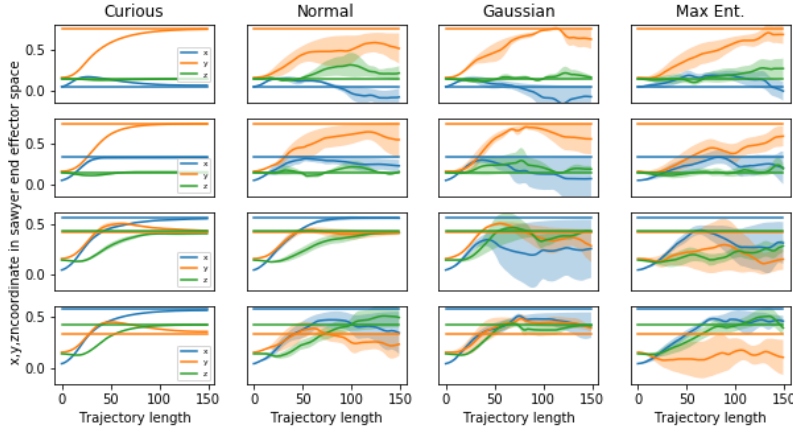## 5.2 Optimizing towards new targets after model learning

Figure 6: Optimizing to reach new targets with regular iLQR after models were learned. 4 different targets (one per row) are evaluated and the final end-effector trajectories presented. Constant lines are targets for x/y/z.

To confirm the hypothesis that the models learned by MBRL with curious iLQR generalize better, because they have explored the state space better, we decided to evaluate the learned dynamics models on a second set of experiments in which the robot tries to reach new, unseen targets. In this experiment we take the GP models learned during experiment 1 in Section 5.1 and use them to optimize trajectories to reach new targets that were not seen during training of the model. The results are shown in Figure 6, where four randomly chosen targets were set and the trajectory was optimized with regular iLQR. Note, that here we use regular iLQR to optimize for the trajectory so that we can better compare the models learned with/without curiosity in the previous set of experiments. Figure 6 shows the trajectory in end effector space for each coordinate dimension, together with the target end effector position as a solid horizontal line. The results are averaged across 5 trials. The trials correspond to using one of the 5 dynamics models at the end of Experiment 1 in Section 5.1. For each trial, the initial torque trajectory was initialized randomly, and the initial joint configuration slightly perturbed. The mean and the standard deviation of the optimized trajectories are computed across the 5 models learned via MBRL with curious iLQR (first col), MBRL with normal iLQR (second col), iLQR with random exploration (third col) and iLQR with maximum entropy exploration bonus (fourth col.). We see that MBRL with curious iLQR results in a model that performs better when presented with a new target. The new targets are reached more reliably and precisely.

6 Real hardware experiments

Target	Distance to Target in m (Learning Iteration)							
	Curious				Normal			
1	0.05 (6)	0.09 (2)	0.09 (3)	**0.07 (3.67)**	0.37 (8)	0.08 (2)	0.18 (8)	**0.21 (7.0)**
2	0.05 (3)	0.09 (4)	0.09 (4)	**0.07 (3.67)**	0.20 (8)	0.08 (3)	0.09 (5)	**0.12 (5.3)**
3	0.09 (6)	0.09 (4)	0.09 (3)	**0.09 (4.33)**	0.17 (8)	0.16 (8)	0.11 (8)	**0.15 (8.0)**
4	0.04 (2)	0.07 (2)	0.07 (2)	**0.06 (2.33)**	0.04 (3)	0.08 (3)	0.05 (3)	**0.06 (3.0)**
				0.07 (3.5)				**0.14 (5.9)**

Table 1: Results on a reaching task. Each task (target) was repeated three times. The mean values are reported in bold font.

	Reaching Precision (m)	
Target	Curious	Normal
1	0.20	0.67
2	0.26	0.61
3	0.25	1.06
4	0.24	0.67
5	0.37	0.49
	0.26	**0.7**

Table 2: Reaching a new target not seen during training.

The experimental platform for our hardware experiments is the Sawyer Robot [29]. The purpose of the experiments was to demonstrate the applicability and the benefits of our algorithm on real hardware. We perform reaching experiments for 4 different target locations. Each experiment is started from scratch with no prior data, and the number of hardware experiments needed to reach the target are compared. The results are summarized in Table 1 and show the number of learning iterations needed in order to reach the target together with the precision in end-effector space. If the target was reached with a precision of below 10 cm, we would consider the task as achieved; if the target was not reached after the 8th learning iteration we would stop the experiment and consider the last end-effector position. We decided to terminate our experiments after the eight iteration as running the experiment on hardware was a lengthy process, as the GP training and the rollout would happen iteratively and GP training time increases with growing amount of data. Also, the reaching precision that we were able to achieve on hardware was significantly lower, compared to the simulation experiments.
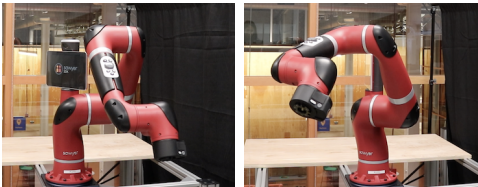
(a) start configuration   (b) target configuration

Figure 7: Joint configuration of Sawyer.

We believe this is due to the data collected from the Sawyer robot, as we could only control the robot at $100Hz$ which introduces inaccuracies when reading the effects of the sent torque command. We repeated each experiment three times to demonstrate the repeatability of our method as we expected measurement noise to affect solutions. From the table we can see that MBRL with curious iLQR would reach a target on average after 3.5 iterations with an average precision of 7 cm, compared to MBRL with regular iLQR that needed 5.9 iterations (often not ever reaching the target after eight iterations with the desired precision), with a precision of 14cm on average. As in simulation, similar to Experiment 5.2 we wanted to evaluate the quality of the learned models on new target positions. The results are summarized in Table 2 and are similar to what we observe in simulation: the models learned with curiosity, when used to optimize for new targets, can achieve higher precision than when using the models learned without curiosity.

7 Conclusion and future work

In this work, we presented a model-based reinforcement learning algorithm that uses an optimal control framework to trade-off between optimizing for a task specific cost and exploring around a locally optimal trajectory. Our algorithm explicitly encourages actions that seek out uncertainties in our model by incorporating them into the cost. By doing so, we are able to learn a model of the dynamics that achieves the task faster than MBRL with standard iLQR, and also transfers well to other tasks. We present experiments on a Sawyer robot in simulation and on hardware. In both sets of experiments, MBRL with *curious iLQR* (our approach) not only learns to achieve the specified task faster, but also generalizes to new tasks and initial conditions. All this points towards the conclusion that resolving dynamics uncertainty during model-based reinforcement learning is indeed a powerful tool. As [2] states, curiosity is a superficial affection: it can arise, diverge and end promptly. We were able to observe similar behaviour in our experiments as well, as can be seen in Figure 5: towards the end of learning, the exploration signal around the trajectory decreases and the robot would explore, deviate from the task slightly, before going back to exploiting once it is fairly certain about the dynamics. In the future, we would like to explore this direction by considering how to maintain exploration strategies. This could be helpful if the robot is still certain about a task, even though the environment or task has changed.

References

[1] C. G. Atkeson and J. C. Santamaria. A comparison of direct and model-based reinforcement learning. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 3557–3564. IEEE, 1997.

[2] G. Loewenstein. The psychology of curiosity: A review and reinterpretation. *Psychological bulletin*, 116(1):75, 1994.

[3] R. W. White. Motivation reconsidered: The concept of competence. *Psychological review*, 66 (5):297, 1959.

[4] J. Kagan. Motives and development. *Journal of Personality and Social Psychology*, 22(1): 51–66, 1972.

[5] Y. Tassa, N. Mansard, and E. Todorov. Control-limited differential dynamic programming. *IEEE International Conference on Robotics and Automation, ICRA*, 2014.

[6] D. H. Jacobson. Optimal Stochastic Linear Systems with Exponential Performance Criteria and Their Relation to Deterministic Differential Games. *IEEE Transaction on Automatic Control*, 18, 1973.

[7] F. Farshidian and J. Buchli. Risk sensitive, nonlinear optimal control: Iterative linear exponential-quadratic optimal control with gaussian noise. *arXiv preprint arXiv:1512.07173*, 2015.

[8] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *NeurIPS*, 2018.

[9] S. Levine and V. Koltun. Guided policy search. In *International Conference on Machine Learning*, pages 1–9, 2013.

[10] S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems*, pages 1071–1079, 2014.

[11] M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011.

[12] S. Schaal. Learning from demonstration. In *Advances in neural information processing systems*, pages 1040–1046, 1997.

[13] M. P. Deisenroth. *Efficient reinforcement learning using Gaussian processes*, volume 9. KIT Scientific Publishing, 2010.

[14] A. G. Barto. Intrinsically motivated learning of hierarchical collections of skills. *International Conference on Developmental Learning and Epigenetic Robotic*, pages 112–119, 2004.

[15] S. Singh, A. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. *18th Annual Conference on Neural Information Processing Systems (NIPS)*, 2004. ISSN 1943-0604.

[16] S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg. Intrinsically Motivated Reinforcement Learning: An Evolutionary Perspective. *IEEE Transactions on Autonomous Mental Development*, 2 (2):70–82, 2010. ISSN 1943-0604.

[17] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479, 2016.

[18] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-Driven Exploration by Self-Supervised Prediction. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2017-July:488–489, 2017. ISSN 21607516.

[19] D. Tanneberg, J. Peters, and E. Rueckert. Intrinsic motivation and mental replay enable efficient online adaptation in stochastic recurrent networks. *Neural Networks*, 109:67–80, 2019.

[20] A. Laversanne-Finot, A. Péré, and P.-Y. Oudeyer. Curiosity driven exploration of learned disentangled goal spaces. *arXiv preprint arXiv:1807.01521*, 2018.

[21] P. Shyam, W. Jaskowski, and F. Gomez. Model-based active exploration. *arXiv preprint arXiv:1810.12162*, 2018.

[22] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[23] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. Rehg, B. Boots, and E. Theodorou. Information theoretic MPC for model-based reinforcement learning. In *ICRA*, 2017.

[24] J. Boedecker, J. T. Springenberg, J. Wülfing, and M. Riedmiller. Approximate real-time optimal control based on sparse gaussian process models. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–8. IEEE, 2014.

[25] P. Whittle. Risk-Sensitive Linear-Quadratic-Gaussian Control. *Advances in Applied Probability*, 13(4):764–777, 1981.

[26] B. Ponton, S. Schaal, and L. Righetti. Risk sensitive nonlinear optimal control with measurement uncertainty. *CoRR*, 2016.

[27] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[28] GPy. GPy: A gaussian process framework in python. http://github.com/SheffieldML/GPy, since 2012.

[29] Sawyer. Rethink robotics. https://www.rethinkrobotics.com/sawyer/, since 2012.

[30] Pybullet. a python module for physics simulation in robotics, games and machine learning. http://pybullet.org/, since 2012.

[31] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.