

# Probabilistic Reinforcement Learning for Control

Marc Deisenroth



Robotics and State Estimation Lab  
University of Washington

June 09, 2008

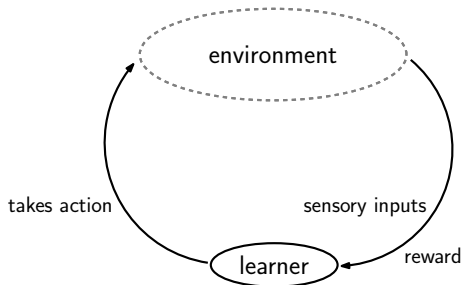
joint work with Carl Edward Rasmussen<sup>1</sup>

<sup>1</sup> Department of Engineering, University of Cambridge, Cambridge, UK

# Motivation

- make a black-box dynamic system achieving a specific task
  - known initial state
  - known goal state
  - interaction with system is expensive
  - solve the task without using too restrictive assumptions
- consider the problem as a **reinforcement learning** problem

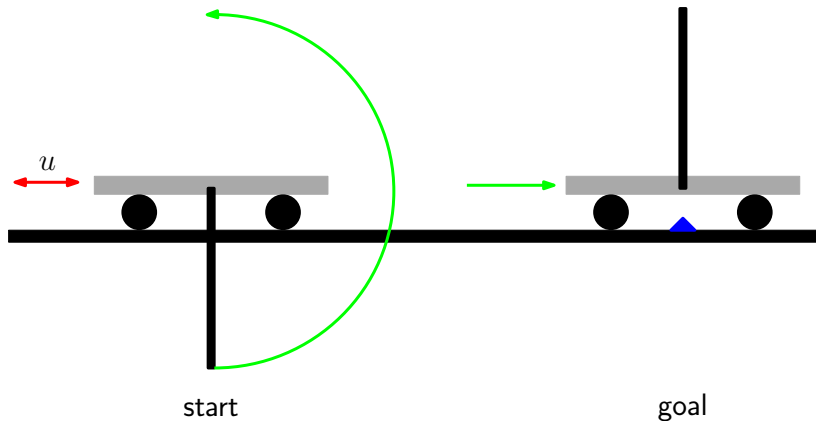
# Reinforcement Learning



→ maximize expected rewards over a long term

# Running Example

cart-pole problem:



→ demo

# Optimal Control Problem

given: initial state distribution  $p(\mathbf{s}_0)$

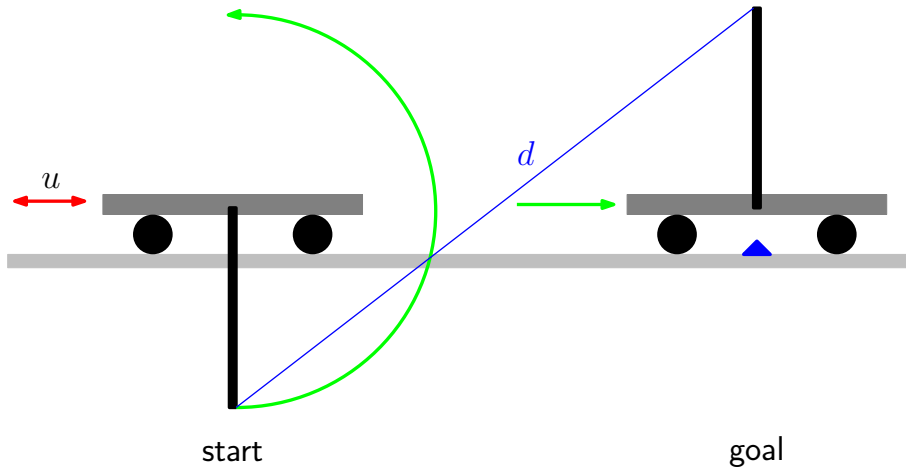
→ find an optimal policy (mapping from states to actions) that minimizes the expected long-term cost

$$\sum_{t=0}^T \mathbb{E} [\ell(\mathbf{s}_t)]$$

challenges:

- minimization
- expectation
- nonlinearities

# Geometric Immediate Cost



$$\ell(s) = 1 - \exp(-c d^2)$$

# Path

starting from  $p(\mathbf{s}_0)$  determine the expected value of a path following a specific policy  $\pi$

## Path evaluation

path:  $\tau = (\mathbf{s}_0, \pi(\mathbf{s}_0), \mathbf{s}_1, \pi(\mathbf{s}_1), \dots, \mathbf{s}_N)$

expected value of path:  $V^\pi(\tau) = \mathbb{E}_\tau \left[ \sum_{t=0}^T \ell(\mathbf{s}_t) \middle| \pi \right]$

# Path

starting from  $p(\mathbf{s}_0)$  determine the expected value of a path following a specific policy  $\pi$

## Path evaluation

path:  $\boldsymbol{\tau} = (\mathbf{s}_0, \pi(\mathbf{s}_0), \mathbf{s}_1, \pi(\mathbf{s}_1), \dots, \mathbf{s}_N)$

expected value of path:  $V^\pi(\boldsymbol{\tau}) = \mathbb{E}_{\boldsymbol{\tau}} \left[ \sum_{t=0}^T \ell(\mathbf{s}_t) \middle| \pi \right]$

if direct interaction with system is **expensive**

→ Monte Carlo path sampling is out of the question

→ need for

- predictions (probabilistic dynamics model)
- uncertainty propagation

# A Classical Control Approach

LQR:

- linearize the (known) system function
- typically high sampling rate
- quadratic cost
- analytic solution

possible problems with LQR:

- can only give a linear controller
- crucial dependence on system dynamics

# A Classical Control Approach

LQR:

- linearize the (known) system function
- typically high sampling rate
- quadratic cost
- analytic solution

possible problems with LQR:

- can only give a linear controller
- crucial dependence on system dynamics

if the system function is unknown:

- system identification: determine parameters of an idealized physical system
- requires a lot of expert knowledge and assumptions
- hand-crafted

# A Machine Learning Approach to Control

interested in

- ▶ (not necessarily linear) **controller** that can achieve the task (not necessarily optimally)
- ▶ something that **models the input-output behavior** of the dynamic system
  - ➔ regression problem:
    - treat everything happening in between as a black box
      - ➔ no explicit parameter fitting (friction, delays, etc.), but also **no** direct relationship between model and underlying physics
    - learn the transition dynamics by observing the system

# A Machine Learning Approach to Control

interested in

- ▶ (not necessarily linear) **controller** that can achieve the task (not necessarily optimally)
- ▶ something that **models the input-output behavior** of the dynamic system
  - regression problem:
    - treat everything happening in between as a black box
      - no explicit parameter fitting (friction, delays, etc.), but also **no** direct relationship between model and underlying physics
    - learn the transition dynamics by observing the system

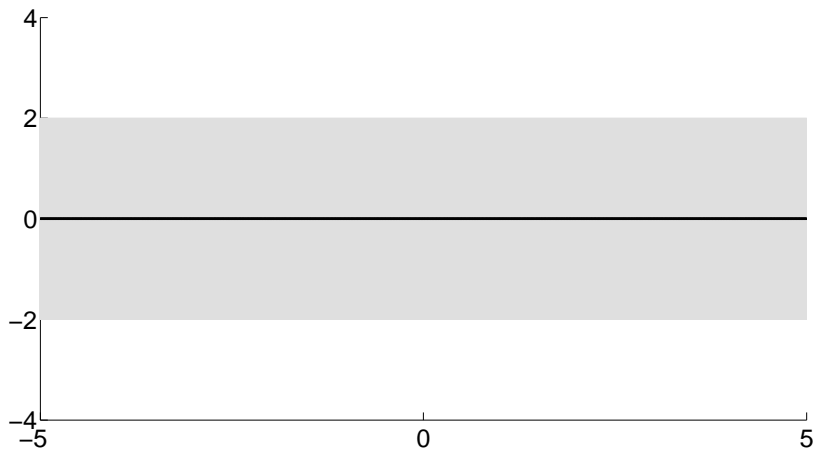
here: utilize **Gaussian process** framework (non-parametric, probabilistic)

# Bayesian Regression with Gaussian Processes

example: model a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  **only** based on (noisy) observations

# Bayesian Regression with Gaussian Processes

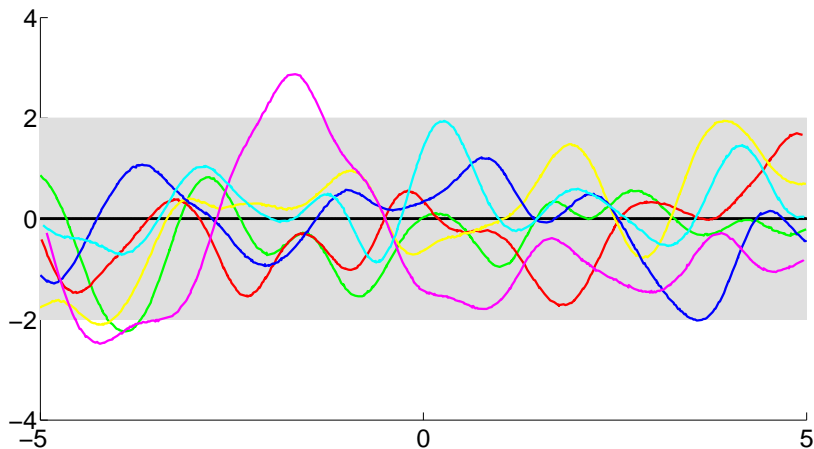
example: model a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  **only** based on (noisy) observations



shaded area: uncertainty about underlying function

# Bayesian Regression with Gaussian Processes

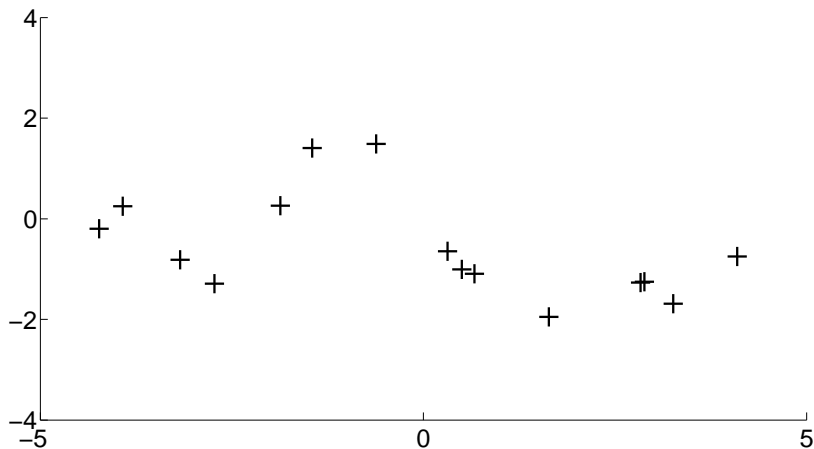
example: model a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  **only** based on (noisy) observations



shaded area: uncertainty about underlying function

# Bayesian Regression with Gaussian Processes

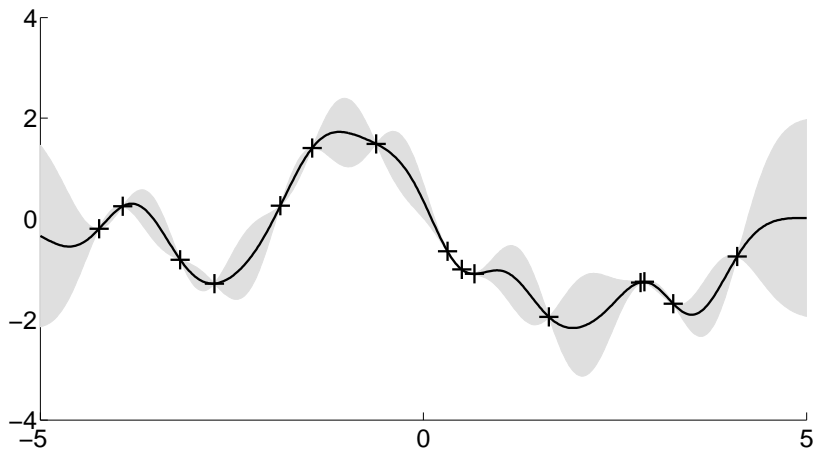
example: model a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  **only** based on (noisy) observations



shaded area: uncertainty about underlying function

# Bayesian Regression with Gaussian Processes

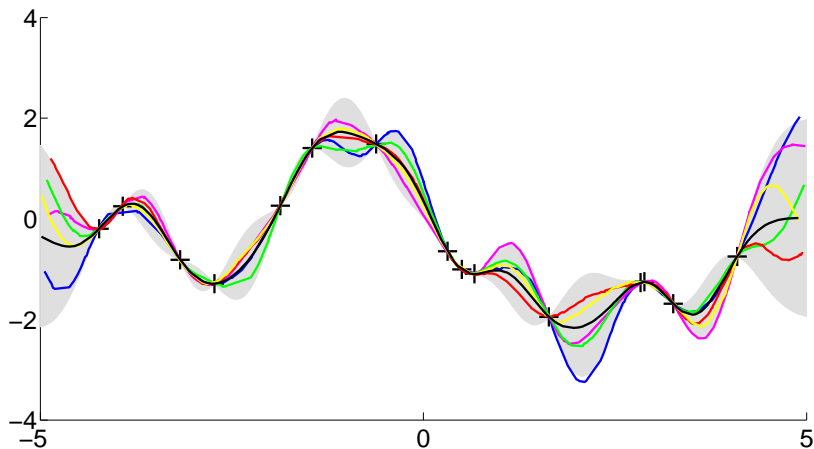
example: model a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  **only** based on (noisy) observations



shaded area: uncertainty about underlying function

# Bayesian Regression with Gaussian Processes

example: model a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  **only** based on (noisy) observations



shaded area: uncertainty about underlying function

# Modeling the Dynamics

- state of cart-pole system is given by position, velocity, angle, angular velocity:  $\mathbf{s} = [x, \dot{x}, \varphi, \dot{\varphi}]^T$
- model a discrete-time system with continuous-valued states and actions:  $\mathbf{s}_{t+1} = f(\mathbf{s}_t, u_t)$ , where  $f$  is unknown
- describe dynamics by 4 GP models (one for each state component):

$$s_{t+\Delta_t}^i - s_t^i \sim \mathcal{GP}(\mathbf{s}_t, u_t), \quad i = 1, \dots, 4$$

- GPs predict **distribution** of successor state, explicitly accounting for uncertainties in learned model and observations
- easy to learn for small  $\Delta_t$ , very difficult for large  $\Delta_t$

# Long-Term Predictions

how to get long-term predictions?

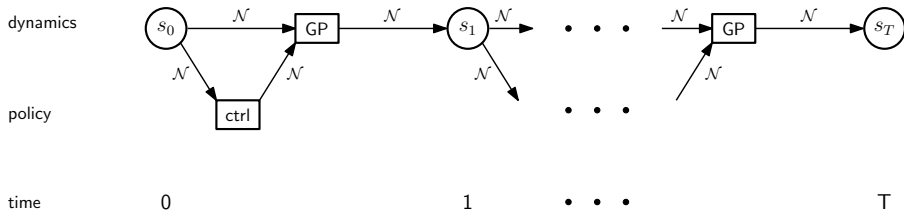
- **cascade** GP predictions (encodes that dynamics are time-invariant)
- keep track of uncertainty evolution → explicitly **propagate uncertainty**
- doable for GPs (moment matching)

(Girard et al., 2003), (Rasmussen and Ghahramani, 2003), (Kuss, 2006)

# Controller Properties

- controller implements a **deterministic** function from states to control signals
- treatment of uncertain inputs (distribution over state)
  - ➔ returns a **distribution over controls**

# Interaction of Controller and Dynamics Model

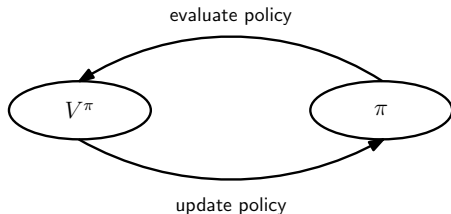


- controller yields distribution over actions for any input distribution
- distribution of successor state is determined based on fully joint Gaussian of current state and control signal
  - control and state variables **covary**
- **simulate** system behavior for a given policy function (**implicit** MC sampling)

# Optimizing Controller Parameters

so far: for a given cost function, we can evaluate the quality of the controller when starting from a set of (distributions of) start states using a fixed policy

- optimal control setting: minimize expected long-term cost
- go for policy iteration-like optimization scheme:



- update of policy/controller parameters through **gradient-based** method

# GP Controller

involved parameters

- **hyperparameters** of the covariance function
- **pseudo training set**: find best function values for given inputs

→ **learn** controller by finding parameters that minimize expected long-term cost (via gradients)

# GP Controller

involved parameters

- **hyperparameters** of the covariance function
- **pseudo training set**: find best function values for given inputs

→ **learn** controller by finding parameters that minimize expected long-term cost (via gradients)

how can we know where the (pseudo) inputs of the GP controller are?

→ optimize these ones as well!

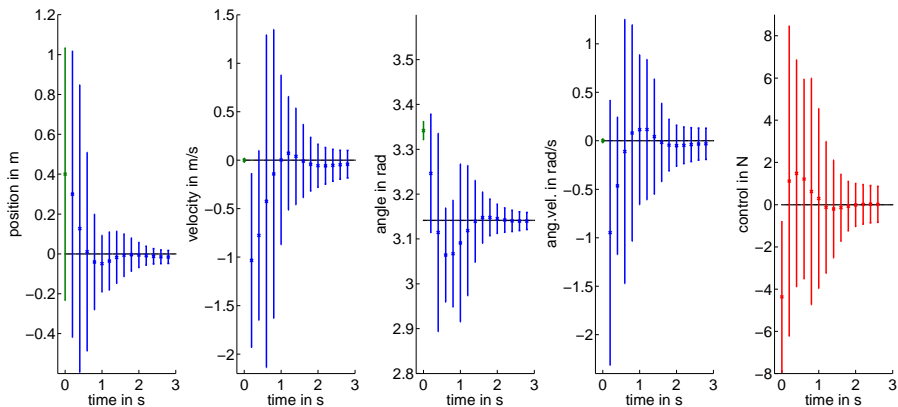
→ training the GP controller relies on **internal simulation** (**no direct interaction** with plant required)

# Experimental Setup

- dynamics are learned offline by 40 seconds (200 training points) of interaction with the system (applying random forces)
- control is limited to a range  $[-u_{\max}, u_{\max}]$

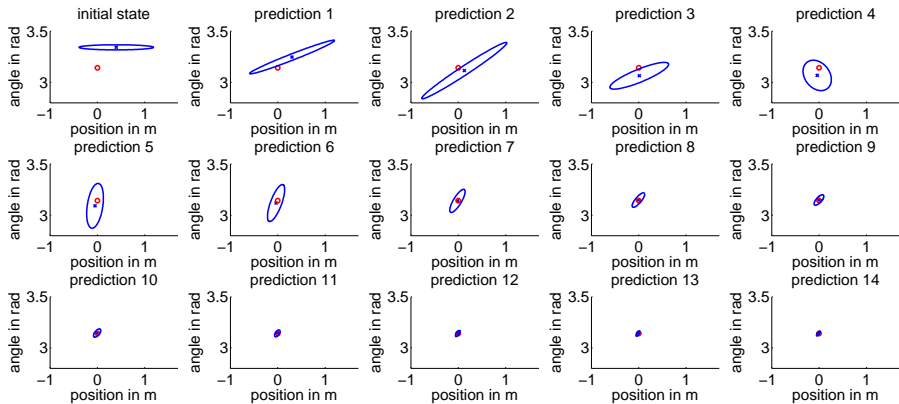
→ demo

# Balancing Task (1)



- high initial uncertainty in position, very small in all other variables
- offset in position and angle

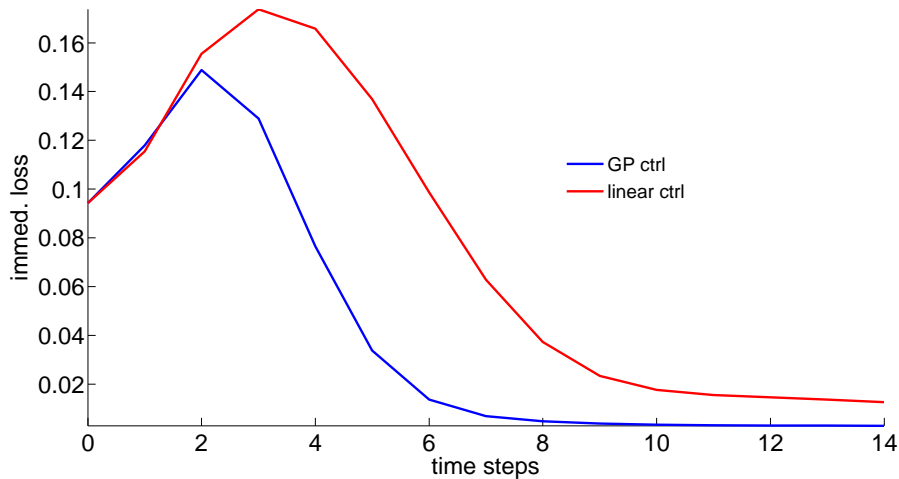
# Balancing Task (2)



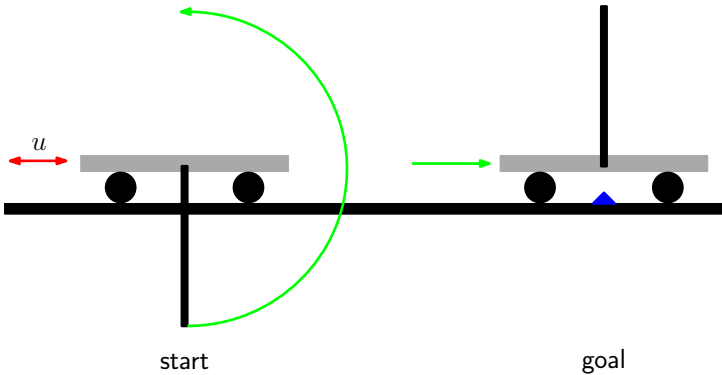
→ demo

# Balancing Task (3)

trajectory pred. immediate losses



# Swing Up Task



→ demo

# Overall Assumptions

required

- distribution of the start states
- some immediate cost (very general in our case)
- here: time-invariant dynamics

# Overall Assumptions

required

- distribution of the start states
- some immediate cost (very general in our case)
- here: time-invariant dynamics

but **not**

- known transition dynamics
- hand-crafted cost function injecting problem-specific properties
- linearization
- explicit Monte Carlo simulation

# GPs for Robust Control?

uncertain model of transition dynamics

- **control**: minimax/ $\mathcal{H}_\infty$  control derives conservative controller for the uncertain dynamics model
- **ML**: GP dynamics model considers  $\infty$  many dynamics models **simultaneously**  
→ final policy is optimal for all  $\infty$  many models (averaging over models)

(Murray-Smith and Sbarbaro, 2002)

# Wrap-up and Future Work

- ▶ nonlinear controller can be **learned**
- ▶ few (and general) assumptions made (in part.: unknown dynamics)
- ▶ fully probabilistic simulation avoids Monte Carlo
- ▶ very **little interaction** with plant
- ▶ maybe also interesting for adaptive and robust optimal control

# Wrap-up and Future Work

- ▶ nonlinear controller can be **learned**
  - ▶ few (and general) assumptions made (in part.: unknown dynamics)
  - ▶ fully probabilistic simulation avoids Monte Carlo
  - ▶ very **little interaction** with plant
  - ▶ maybe also interesting for adaptive and robust optimal control
- 
- ▶ learn transition dynamics online
  - ▶ application to a real system

# References



Agathe Girard, Carl E. Rasmussen, Joaquin Quiñero Candela, and Roderick Murray-Smith.

**Gaussian Process Priors with Uncertain Inputs—Application to Multiple-Step Ahead Time Series Forecasting.**

In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 529–536. The MIT Press, Cambridge, MA, USA, 2003.



Malte Kuss.

**Gaussian Process Models for Robust Regression, Classification, and Reinforcement Learning.**

PhD thesis, Technische Universität Darmstadt, Germany, February 2006.



Roderick Murray-Smith and Daniel Sbarbaro.

**Nonlinear Adaptive Control Using Non-Parametric Gaussian Process Prior Models.**

In *Proceedings of the 15th IFAC World Congress*, volume 15, Barcelona, Spain, July 2002. Academic Press.



Carl E. Rasmussen and Zoubin Ghahramani.

**Bayesian Monte Carlo.**

In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 489–496. The MIT Press, Cambridge, MA, USA, 2003.



Carl E. Rasmussen and Christopher K. I. Williams.

**Gaussian Processes for Machine Learning.**

Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 2006.