Robot Learning

Winter Semester 2020/2021, Homework 5

Prof. Dr. J. Peters, J. Watson, J. Carvalho, J. Urain and T. Dam

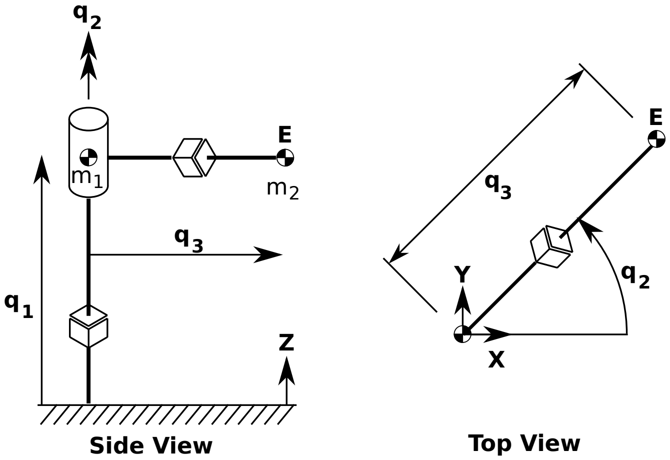
TECHNISCHE
UNIVERSITÄT
DARMSTADT

Total points: 97
Due date: Monday, 22 February 2021 (23:59)

Name, Surname, ID Number

Problem 5.1 Model Learning [28 Points]

The new and improved Spinbot 2000 is a multi-purpose robot platform. It is made of a kinematic chain consisting of a linear axis q_1, a rotational axis q_2 and another linear axis q_3, as shown in the figure below. These three joints are actuated with forces and torques of u_1, u_2, and u_3. Different end effectors, including a gripper or a table tennis racket, can be mounted on the end of the robot, indicated by the letter E. Thanks to Spinbot's patented SuperLight technology, the robot's mass is distributed according to one point mass of m_1 at the second joint and another point mass of m_2 at the end of the robot E.



Side View

Top View

The inverse dynamics model of the Spinbot is given as

$$
\begin{aligned}
u_1 &= (m_1 + m_2)(\ddot{q}_1 + g),\\
u_2 &= m_2\left(2\dot{q}_3\dot{q}_2 q_3 + q_3^2\ddot{q}_2\right),\\
u_3 &= m_2\left(\ddot{q}_3 - q_3\dot{q}_2^2\right).
\end{aligned}
$$

We now collected 100 samples from the robot while using a PD controller with gravity compensation at a rate of 500Hz. The collected data (see `spinbotdata.txt`) is organized as follows

	t_1	t_2	t_3	...
$q_1[m]$				
$q_2[rad]$				
$q_3[m]$				
...				
$\ddot{q}_3[m/s^2]$				
$u_1[N]$				
$u_2[Nm]$				
$u_3[N]$				

Given this data, you want to learn the inverse dynamics of the robot to use a model-based controller. The inverse dynamics of the system will be modeled as $u_i = \boldsymbol{\phi}_i(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})^\top \boldsymbol{\theta}_i$, for $i = 1, 2, 3$, where $\boldsymbol{\phi}_i$ are features and $\boldsymbol{\theta}_i$ are the parameters.

a) Problem Statement [2 Points]

What kind of machine learning problem is learning an inverse dynamics model? What kind of information do you need to solve such a problem?

b) Assumptions [5 Points]

Which standard assumption has been violated by taking the data from trajectories?

c) Features and Parameters [4 Points]

Assuming that the gravity g is unknown, what are the feature vectors $\boldsymbol{\phi}_i$ and the corresponding parameter vectors $\boldsymbol{\theta}_i$ for each torque? (Hint: you do not need to use the data at this point)

d) Learning the Parameters [2 Points]

You want to compute the parameters $\boldsymbol{\theta}_i$ by minimizing the squared error between the estimated torques and the actual torques. Write down the linear algebra equation that you would use to compute the parameters. For each torque, write down the dimensions of each matrix in the equation.
Then, compute the least-squares estimate of the parameters $\boldsymbol{\theta}_i$ from the data and report the learned values for each output.

e) Recovering Model Information [4 Points]

Can you recover the mass properties m_1 and m_2 from your learned parameters? Has the robot learned a plausible inverse dynamics model? Does using the true gravitational constant g make a difference? Explain your answers.

f) Model Evaluation [7 Points]

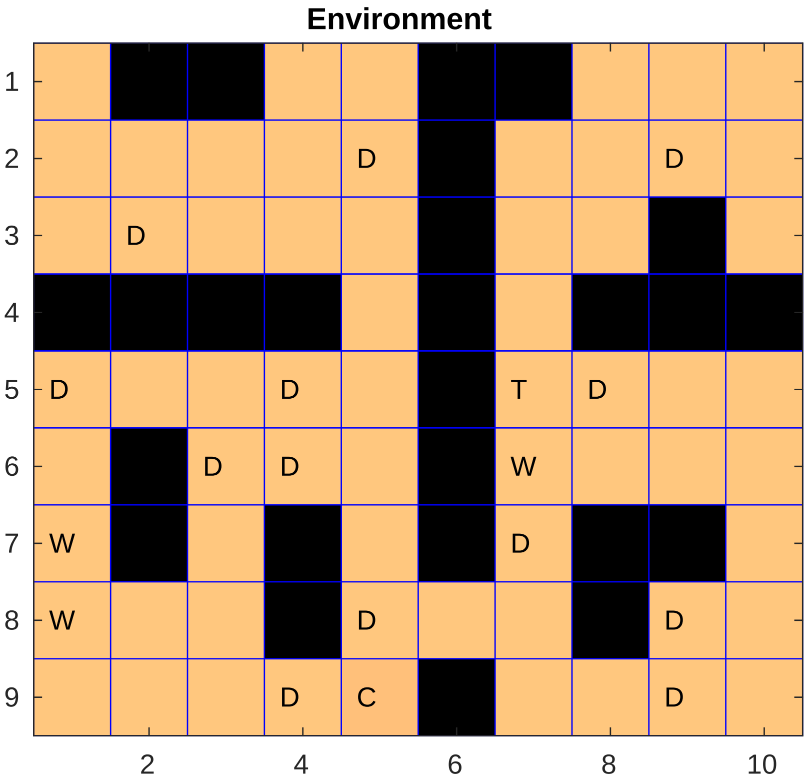
Plot the forces and torques predicted by your model over time, as well as those recorded in the data and comment the results. Is the model accuracy acceptable? If not, how would improve your model? Use one figure per joint.

g) Models for Robot Learning [4 Points]

Name and describe three models we may wish to learn for robotics and one or more possible application.

Problem 5.2 Reinforcement Learning [34 Points]

You recently acquired a robot for cleaning you apartment but you are not happy with its performance and you decide to reprogram it using the latest AI algorithms. As a consequence the robot became self-aware and, whenever you are away, it prefers to play with toys rather than cleaning the apartment. Only the cat has noticed the strange behavior and attacks the robot. The robot is about to start its day and its current perception of the environment is as following



The black squares denote extremely dangerous states that the robot must avoid to protect its valuable sensors. The reward of such states is set to $r_{\text{danger}} = -10^5$ (NB: the robot can still go through these states!). Moreover, despite being waterproof, the robot developed a phobia of water (W), imitating the cat. The reward of states with water is $r_{\text{water}} = -100$. The robot is also afraid of the cat (C) and tries to avoid it at any cost. The reward when encountering the cat is $r_{\text{cat}} = -3000$. The state containing the toy (T) has a reward of $r_{\text{toy}} = 1000$, as the robot enjoys playing with them. Some of the initial specification still remain, therefore the robot receives $r_{\text{dirt}} = 35$ in states with dirt (D).

The reward collected at an instant of time “t” depends only on the state where the robot is at that instant. Assume that the robot collects the reward at exactly the same instant it starts executing an action and that each action takes one time step to be executed. The robot can perform the following actions: down, right, up, left and stay.

In our system we represent the actions with the an ID (0, 1, 2, 3, 4), while the grid is indexed as `{row,column}`. The robot can’t leave the grid as it is surrounded with walls. A skeleton of the gridworld code and some plotting functions are available in Moodle.

a) Finite Horizon Problem [10 Points]

In the first exercise we consider the finite horizon problem, with horizon $T = 15$ steps. The goal of the robot is to maximize the expected return

$$J_\pi = \mathbb{E}_\pi \left[\sum_{t=1}^{T-1} r_t \left(s_t, a_t\right) + r_T \left(s_T\right) \right], \tag{1}$$

according to policy π, state s, action a, reward r, and horizon T. Since rewards in our case are independent of the action and the actions are deterministic, Equation (1) becomes

$$J_\pi = \sum_{t=1}^{T} r_t \left(s_t\right). \tag{2}$$

Using the Value Iteration algorithm, determine the optimal action for each state when the robot has 15 steps left. Attach the plot of the policy to your answer and a mesh plot for the value function. Describe and comment the policy: is the robot avoiding the cat and the water? Is it collecting dirt and playing with the toy? With what time horizon would the robot act differently in state $(9,4)$?

b) Infinite Horizon Problem - Part 1 [4 Points]

We now consider the infinite horizon problem, where $T = \infty$. Rewrite Equation (1) for the infinite horizon case adding a discount factor γ. Explain briefly why the discount factor is needed.

c) Infinite Horizon Problem - Part 2 [6 Points]

Calculate the optimal actions with the infinite horizon formulation. Use a discount factor of $\gamma = 0.8$ and attach the new policy and value function plots. What can we say about the new policy? Is it different from the finite horizon scenario? Why?

d) Finite Horizon Problem with Probabilistic Transition Function [10 Points]

After a fight with the cat, the robot experiences control problems. For each of the actions up, left, down, right, the robot has now a probability 0.7 of correctly performing it and a probability of 0.1 of performing another action according to the following rule: if the action is left or right, the robot could perform up or down. If the action is up or down, the robot could perform left or right. Additionally, the action can fail causing the robot to remain on the same state with probability 0.1. Using the finite horizon formulation, calculate the optimal policy and the value function. Use a time horizon of $T = 15$ steps as before. Attach your plots and comment them: what is the most common action and why does the learned policy select it?

e) Reinforcement Learning - Other Approaches [4 Points]

What are the two assumptions that let us use the Value Iteration algorithm? What if they would have been not satisfied? Which other algorithm would you have used? Explain it with your own words and write down its fundamental equation.

Problem 5.3 Episodic Policy Search with Policy Gradients [30 Points]

In this exercise your task is to control a 2-DoF planar robot to throw a ball at a specific target. You will use an episodic setup, where you first specify the parameters of the policy, evaluate them on the simulated system, and obtain a reward. The robot will be controlled with the Dynamic Motor Primitives (DMPs). The goal state of the DMPs is pre-specified and the weights of the DMP $\theta_i, i = 1 \dots 10$ are the open parameters of the control policy. Each DoF of the robot is controlled with a DMP with five basis functions. The ball is mounted at the end-effector of the robot and gets automatically released at time step t_{rel}. We define a stochastic distribution $\pi(\boldsymbol{\theta}|\boldsymbol{\omega}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with $\boldsymbol{\omega} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$.
Your task is to update the parameters $\boldsymbol{\omega}$ of the policy to maximize the expected return. In this exercises we will not modify the low-level control policy (the DMPs) of the robot, but rather we will optimize the expected return under the search distribution using policy gradients.

A template for the simulation of the 2-DoF planar robot and plotting functions can be found in Moodle.

a) Analytical Derivation [5 Points]

You have a Gaussian policy with diagonal covariance, i.e., $\pi(\boldsymbol{\theta}|\boldsymbol{\omega}) = \mathcal{N}\left(\boldsymbol{\mu}, \operatorname{diag}\left(\boldsymbol{\sigma}^2\right)\right)$, where $\boldsymbol{\omega} = [\boldsymbol{\mu}, \boldsymbol{\sigma}]$. Compute analytically the gradient of the logarithm of the policy with respect to the parameters $\boldsymbol{\omega}$, i.e., $\nabla_{\boldsymbol{\omega}} \log \pi(\boldsymbol{\theta}|\boldsymbol{\omega})$. (Hint: consider the properties of the diagonal covariance matrix.)

b) Programming Exercise [5 Points]

Optimize the upper-level policy parameters using the policy gradient. Use an initial mean of $\boldsymbol{\mu}_0 = [0 \ldots 0]$ and a fixed $\boldsymbol{\sigma} = \mathrm{diag}([10 \ldots 10])$ (i.e., do not update $\boldsymbol{\sigma}$). Set the learning rate to $\alpha = 0.1$ and use 25 episodes sampled for each iteration and max 100 iterations of policy updates.
Repeat the learning 10 times and plot the mean of the average return of all runs with 95% confidence. Comment your results.

c) A Little Trick [5 Points]

How would you improve the above implementation? (Beside using a smaller or adaptive learning rate, or using the natural gradient). What is the theory behind this “trick”? Repeat the learning and discuss the results.

d) Learning Rate [5 Points]

Repeat the optimization changing the learning rate to $\alpha = 0.4$ and $\alpha = 0.2$ (keep the trick of the previous exercise). Plot in one figure the mean of the average returns for all α with 95% confidence. How does the value of α affect the convergence of the algorithm?

e) Variable Variance [5 Points]

Try to improve the optimization process by learning also the variance $\boldsymbol{\sigma}$. Is it easier or harder to learn also the variance? Why?
Without using the natural gradient, tune the learning process to achieve better results. If you think it is necessary, you can impose a lower bound to avoid that the variance collapses to infinitely small values (e.g., if $\sigma(i) < \sigma_{\text{lower}}$ then $\sigma(i) = \sigma_{\text{lower}}$). In one figure, plot the learning trend with confidence interval as done before and compare it to the one achieved with $\alpha = 0.4$ before.

f) Natural Gradient [5 Points]

Write down the equation of the natural gradient. What is the theory behind it? Is it always easy to use?

Problem 5.4 Reinforcement Learning [5 Points]

a) RL Exploration Strategies [5 Points]

In which spaces can you perform exploration in RL? Discuss the two exploration strategies applicable to RL.