Learning inverse dynamics for a 7 degree of freedom robot arm

Konstantinos Vourloumis, kv810@nyu.edu
Julian Viereck, jv1439@nyu.edu

***Abstract*— When controlling a robot there are many different ways to describe the desired behavior. One common way is to plan a desired trajectory and then compute the accelerations along this path for the robot to follow it. For a torque controlled robot, given the desired accelerations the necessary torques at each actuated joint must be computed. This computation is known as computing the inverse dynamics in the robotics literature. In this work, we compare different machine learning methods to learn the inverse dynamics for a real world 7 degree of freedom robot arm.**

I. Introduction

Over the last years, the degree of automation has increased and with it also the usage of robots. Especially in assembly lines, robots are used to improve the process. A common task that robots have to perform is following a desired trajectory. For instance, when reaching a cup, a robot arm goes from an initial position towards the cup. To make the robot follow this trajectory, a common way is by computing the robot's joint angles $\boldsymbol{q}$ at every timestep. Given these positions, the desired velocity $\dot{\boldsymbol{q}}$ and acceleration $\ddot{\boldsymbol{q}}$ along the trajectory is computed (using differentiation) and the control task reduced to track this trajectory.

For position controlled robots, tracking such a desired trajectory is taken care of by the robots firmware. However, for a torque controlled robot, where the inputs to the system are the desired torques at each actuated joints $\boldsymbol{\tau}$, one has to convert the desired position trajectory into torque commands. This can be done using a proportional-integral-derivative (PID) controller. For such a controller, finding the right gains becomes a problem as lower gains make the robot arm more compliant while reducing the tracking precision. Another way to track the trajectory is by deriving the necessary torques based on a physical model of the system. Given an n-link robot arm, the dynamics of the system can be written as [1]:

$$\boldsymbol{\tau} = \boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}), \tag{1}$$

where $\boldsymbol{H}$ denotes the inertia matrix, $\boldsymbol{C}$ accounts for the centripetal and Coriolis torques and $\boldsymbol{G}$ for the torques due to gravity acting on the system. $\boldsymbol{q}$ are the joint positions of the manipulator, $\dot{\boldsymbol{q}}$ its velocity and $\ddot{\boldsymbol{q}}$ its (desired) acceleration.

For working on a real robot, the torques derived from Equation (1) are often wrong. For instance, the exact inertia matrix $\boldsymbol{H}$ on a real robot is often unknown. In addition, the model from Equation (1) does not incorporate friction and other disturbance effects. A way to overcome this model vs reality mismatch is by using learning techniques to learn the inverse dynamics directly from data. This has been done before, for instance in [2], [3] and [4].

In this project, we compare different machine learning methods for learning an inverse dynamics model. In particular, we are going to use the SARCOS data [5], which was used in previous publications ([6], [7], [8]). The data consists of 44,484 training samples and 4,449 test samples created on an SARCOS anthropomorphic robot arm. Each sample consists of 7 joint position, velocity and acceleration and 7 target joint torques.

As methods, we compare Support Vector Regression, Random Forest Regression and Neural Networks. We compare different models and select the best model using cross validation. The quality of the prediction is measured by using the mean squared error (MSE). As predictors we use the position, velocity and acceleration of a single timestep as well as the current and the last two samples before.

References

[1] R. M. Murray, *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[2] J.-J. E. Slotine and W. Li, "On the adaptive control of robot manipulators," *The international journal of robotics research*, vol. 6, no. 3, pp. 49–59, 1987.

[3] N. Ratliff, F. Meier, D. Kappler, and S. Schaal, "Doomed: Direct online optimization of modeling errors in dynamics," *Big data*, vol. 4, no. 4, pp. 253–268, 2016.

[4] F. Meier, D. Kappler, N. Ratliff, and S. Schaal, "Towards robust online inverse dynamics learning," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 4034–4039.

[5] "The sarcos data," http://www.gaussianprocess.org/gpml/data/, accessed: 2018-04-20.

[6] S. Vijayakumar, "Lwpr: An o (n) algorithm for incremental real time learning in high dimensional space," in *Proc. Seventeenth International Conference on Machine Learning, 2000*, 2000.

[7] S. Vijayakumar, A. D'souza, T. Shibata, J. Conradt, and S. Schaal, "Statistical learning for humanoid robots," *Autonomous Robots*, vol. 12, no. 1, pp. 55–69, 2002.

[8] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural computation*, vol. 17, no. 12, pp. 2602–2634, 2005.