Reinforcement Learning Based Approach For Direct-Drive Robotic Joint Friction Compensation

Matt McDermott

Abstract:

The goal of this project was to develop a policy through Reinforcement Learning that could counteract the state dependent frictional losses present in each joint of a 3DOF back-drivable robotic arm. This paper describes two different attempts to solve this problem using Deep Deterministic Policy Gradients (DDPG). The first attempt involves comparing a ground truth simulated arm to an arm in which the coefficients of friction can be changed by the output layer of an "actor" neural network. If friction parameters can be determined through this strategy, they can be accounted for directly. The alternative strategy follows a more traditional approach which involves directly controlling joint torques in order to direct the arm to a position setpoint. Using information from an initial disturbance, this network could be used to command the arm to move at a constant velocity in the direction in world space of the initial disturbance.

Introduction:

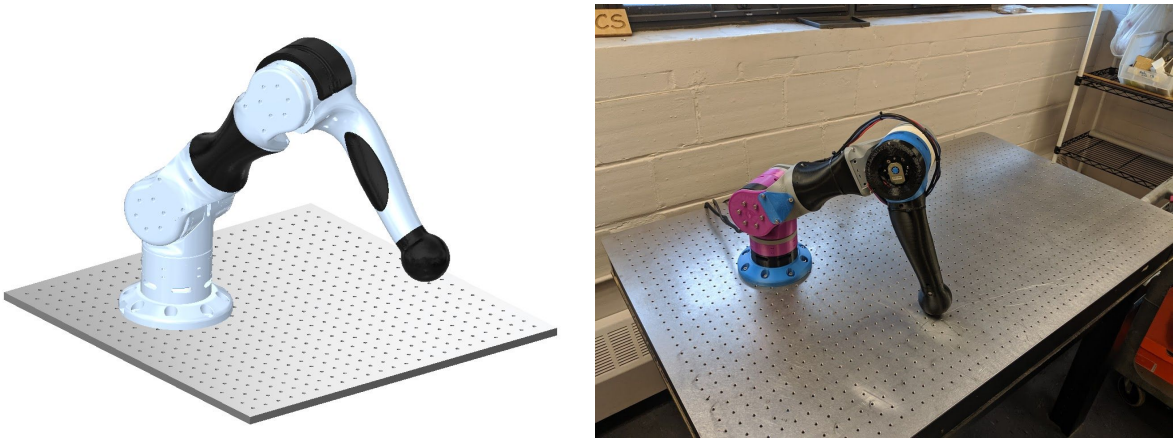


Fig. 1: 3DOF Backdrivable Actuator

Unlike other physical parameters such as mass or the length of a link, friction parameters that govern the dynamics of a robot within a given configuration can not be measured directly. Friction must be determined experimentally. A simple method to cancel out some of the effects of friction is to assume a model of only viscous damping in which the energy lost to friction at a joint is linearly correlated to the velocity of the joint. Parameters may be determined experimentally by adding in torque to the system according to some parameter β multiplied by joint velocity. β is increased in magnitude until just before instability is reached, meaning any disturbance to the joint causes it to accelerate away from its starting

location. This crude strategy can be enough for reducing a large portion of system friction but is insufficient for accurately modeling the full dynamics of the system as it ignores the entire regime of slip-stick friction which is responsible for the behavior of a joint at the onset of movement from a static position. For this reason, three parameters must be considered; Fs, Fk and β representing the forces associated with static and kinetic friction, as well as the constant associated with viscous damping.

Furthermore, due to changes in the loading of joints at different configurations which will change the normal force exerted on mating surfaces, the overall forces of friction are state dependent. For instance, the cantilever effect that occurs when the actuator reaches to points far from the base will increase the radial loading of the bearings on the base joint and may affect friction. Additionally, imperfections in a planetary gearbox may result in periodic changes in resistance. While these effects are present in any robot, accounting for them is especially important in the case of rapid prototyping, as 3D printed plastic joints are especially susceptible manufacturing defects and strain under load.

The state dependant values of the friction constants encode changes in normal force between mating surfaces of joints due to deflection, higher axial loading of base joints as the arm reaches out farther in its workspace, periodic changes in friction due to defects inside gearboxes, the friction of wire harnesses, and other effects that would be incredibly difficult to take into account when deriving equations of motion. For that reason, the values determined for each parameter will represent the product of the coefficients of friction multiplied by whatever the normal force must be at that state.

If a model of joint friction can be developed with sufficient accuracy, a feedforward controller can be implemented which may counteract the forces of friction acting on the system via torque control, thus removing all friction from the system. When implemented in conjunction with gravity and inertia cancellation, this may allow for a much higher degree of transparent actuation. In practice, it is not feasible to cancel all of inertia or friction (as any disturbance would cause the controller to accelerate without bound) though the more of each that can be cancelled, the less the behavior of the end effector is affected by the dynamics of the robot.

Transparent actuation has many applications and is especially useful for tasks in which a human user must continuously work in direct contact with a robotic manipulator. In the case of surgical simulation, making haptic rendering devices more transparent could be less cumbersome for users and provide more realistic behavior which could improve surgeon performance. Implementing transparent actuation in a powered upper body exoskeleton can allow stronger devices to manipulate more massive objects while still remaining relatively transparent in their movements and receptive to human force input.

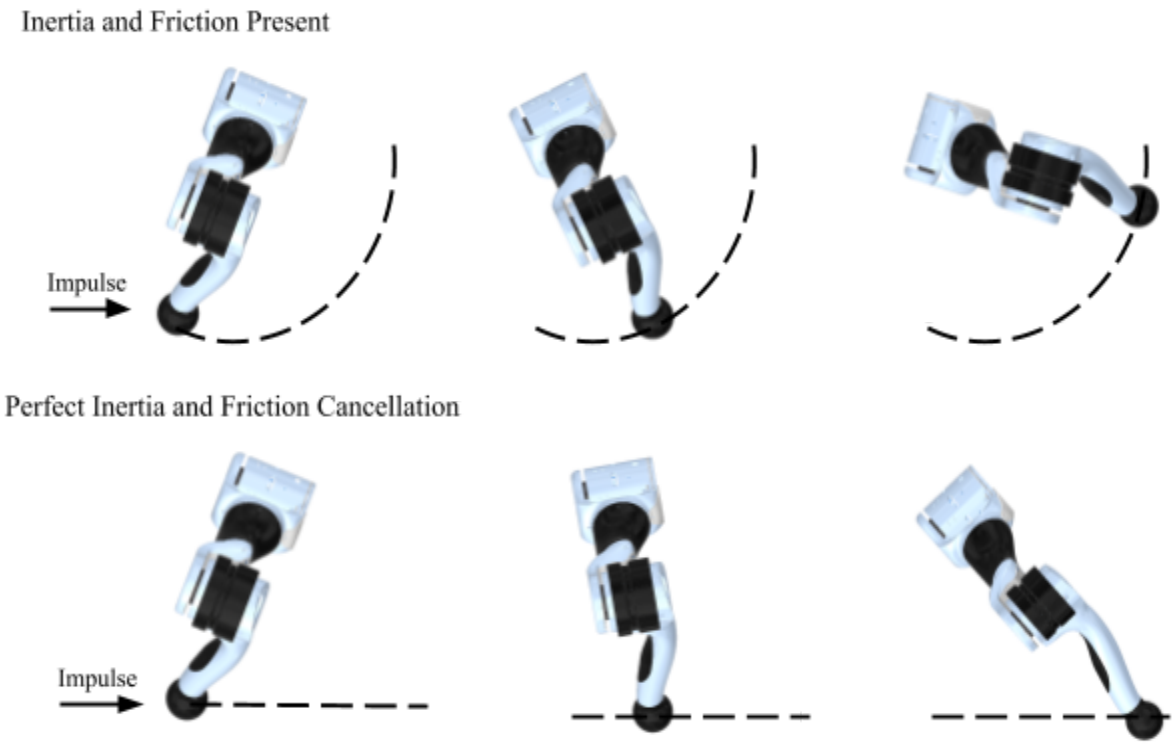


Fig 2: Transparent Actuation

Background and Related Work:

There are numerous techniques used for parameter identification of highly nonlinear systems. One article specifically relating to friction, *Modeling, Identification, and Compensation of Slip-Stick Friction* describes a method of designing an industrial controller to compensate frictional losses that adjusts changes made to the policy based on which regime of friction dominated the movement of the system at the current timestep.

The 1999 paper *Actor-Critic Algorithms* lays the groundwork for much of the network design used in this problem. The 2015 paper *Continuous Control with Deep Reinforcement Learning* builds on these ideas and introduces novel improvements to DDPG, which was used to solve this problem.

Methodology:

Because the task of friction identification involves using both continuous input and output spaces, tabular methods can not be used. DDPG is an off-policy RL technique that concurrently learns a Q-function and an optimal policy [3]. DDPG uses batches of samples chosen from a replay buffer of past state transitions (s,a,r,s',d) to determine how to create a Target which will hold values that the Agent should be close to. One problem associated with vanilla Actor-Critic networks is the fact that Target and local networks rely on the same parameter space which can lead to divergence. To get around this issue,

DDPG slows the updates of the Target Networks to ensure that the parameter sets are not the same. The implementation for this problem used a soft update calculated as:

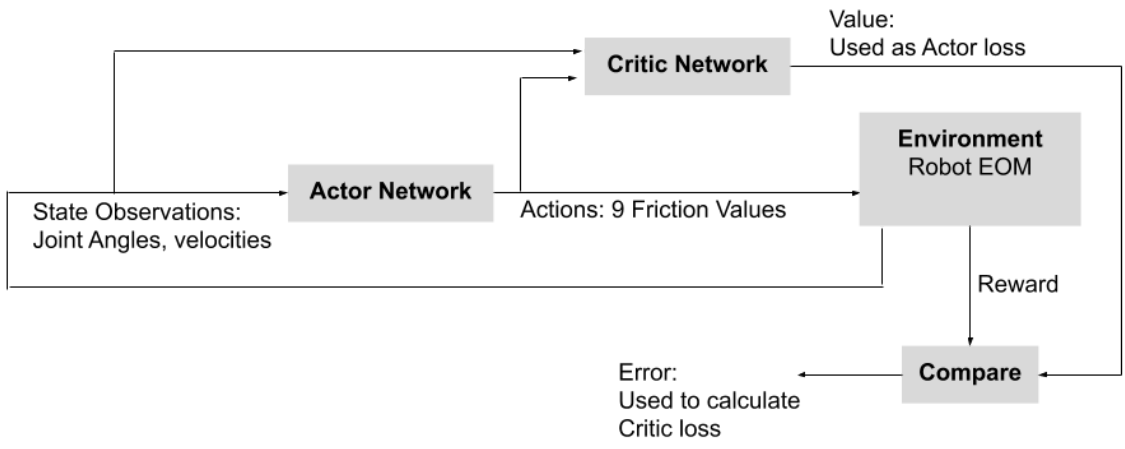
$$\theta_{target} = \tau * \theta_{local} + (1 - \tau) * \theta_{target}$$

In my implementation, the value for this Polyak constant was 0.005, meaning that at each update the target network would be only slightly changed by the current parameters.

Rather than relying on numerical integration, a closed form set of Equations of Motion (EOM) were determined for the system which included contributions from static, kinetic and viscous terms. The advantage of using EOM is that they can be used to obtain a perfectly accurate estimate of future states arbitrarily far into the future without needing to iteratively step forward to obtain a solution. Because the forces on the system needed to be continuously differentiable, the step function that determined the transition between static and kinetic friction was approximated as:

$$\frac{1}{1+e^{-10000x}}$$

In order to simplify the scope of this project, all trials were run in simulation with ground truth friction values estimated by comparing simulations of the arm at different values to videos of the real life arm upside down swinging freely. During training, each trial would begin with randomly initialized starting positions and velocities for each joint. The states of the joints would be fed into the actor network which would output its best estimate for the action that should be taken- the values at which to set joint friction parameters given the current state. These joint friction values would be plugged into the EOM and the next states after 0.1 seconds would be compared against the states of the arm after the same interval if the ground truth values for friction parameters were used. This sum of error squared between these states would be used for the reward function. In the case of configuration dependent friction, the ground truth friction values of the new states would need to be updated before calculating error.



After repeated attempts to develop a model for predicting friction parameters failed to converge, an alternate approach was devised to cancel out the effects of inertia and friction acting on the system by controlling torque directly. Instead of attempting to identify the parameters of friction, an actor agent was

fed in current states of the system as well as three goal joint angles. The agent was rewarded based on how close the joints moved toward the goal positions.

Experimental Results:

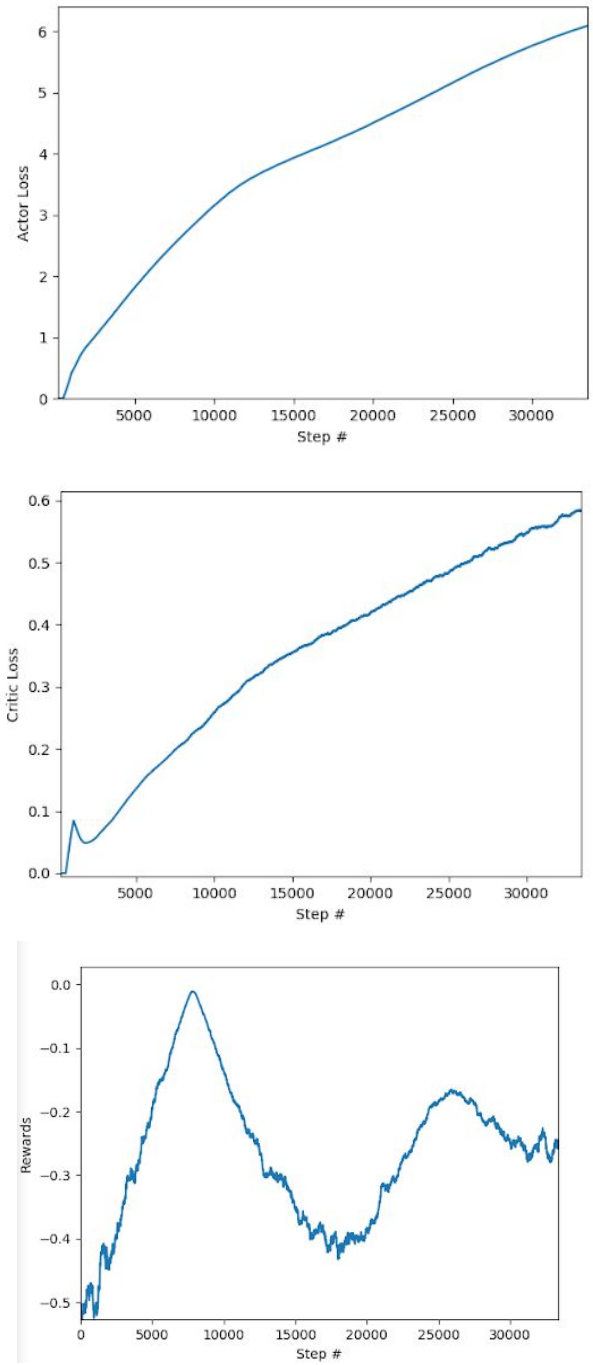


Fig 4: Results of Part I

Unfortunately, various attempts at direct identification of friction parameters were unsuccessful. Models consistently failed to converge and would overshoot ground truth values for each parameter.

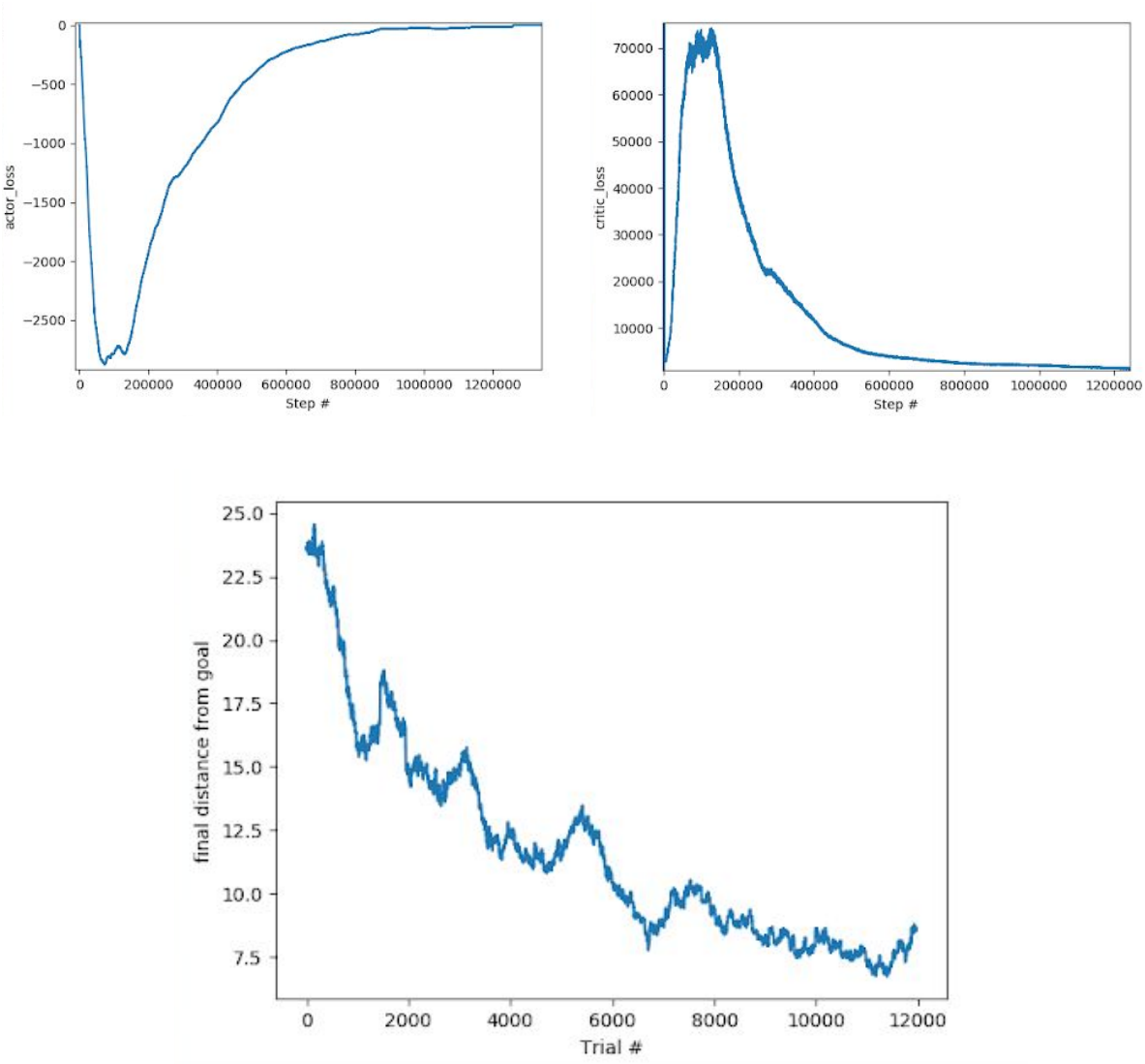


Fig. 5: Results of Part II

While there remains plenty of room for optimization, the second set of networks successfully converged on a policy capable of moving arm from an arbitrary set of starting states to any goal position within the range of -pi to +pi for each joint. Above results were obtained after 20 hours of continuous training on an Nvidia GTX 1060. The third chart displaying final distance from the goal position represents the moving average of the reward for the last 250 trials which is the sum of the squares of the error for each joint.

Conclusion and Future Work:

A potential root cause for the model failing to converge in the first implementation is the fact that the problem is not a true sequential decision making task. Because the reward of each action taken is independent of the reward for the following action, it is possible that there is not enough information being given to the network to determine an optimal policy. While the attempts at directly identifying friction values in this project with the current reward function were unsuccessful, it is possible that this strategy may work if the action of the actor network is redefined as a set of values that are used to slightly increase or decrease variables containing estimates of friction parameters in a given state.

The approach of determining the friction parameters directly remains to be an attractive way to solve this problem because it does not rely as heavily on real world experimental data. A few trials would still be needed to record path data for the ground truth robot, however, because the real arm in this case is merely being used as a benchmark through which to test the accuracy of parameters chosen by the actor agent. A possible remedy for the lack of sequential decision making in this approach could be to create a number of variables to hold values for friction parameters for the environment (input of the actor network) and to have the actions taken be some value to be added or subtracted from the current variable. These variables storing friction values would be updated at each timestep as a result of the changes made in the previous timestep.

Alternatively, changes made to the second approach have the potential to sidestep the issue of system identification entirely and could still result in an agent capable of transparent behavior. If goal velocity setpoints are added to the network (bringing the total input layer to be 12 neurons; 3 input position, 3 input velocity, 3 output position and 3 output velocity) then the agent could be trained to move and reach a certain velocity at a desired location. This could be manipulated to create transparent behavior by instructing the arm to move in a linear path at a constant velocity after measuring the displacement and acceleration of an initial disturbance. While using the equations of motion for friction estimation rather than numerical integration is incredibly useful for the first approach of this problem, due to the fact that torque inputs of the joints were the actions taken (and must be updated at every timestep), the EOM needed to be re-evaluated after every timestep which provided significantly worse performance than numerical integration.

Using Reinforcement Learning for system identification has many powerful use cases. If the strategies of determining friction parameters discussed in this paper can be made to work in simulation, it is likely that they can be extended to work with mechatronic systems in the real world with a relatively small number of training iterations. Identifying key mechanical parameters of rapid prototypes can cut down on development time and manufacturing cost. Furthermore, RL based friction estimation has the potential to push the boundary of how much of a system's dynamics may be cancelled out, leading to higher performing, safer, and more transparent systems.

References:

1. OpenAI, Spinning Up. Deep Deterministic Policy Gradient. Retrieved December 20, 2020, from https://spinningup.openai.com/en/latest/algorithms/ddpg.html?source=post_page

2. Marton, Lrinc. "Modeling, Identification, and Compensation of Stick-Slip Friction." *IEEE Transactions on Industrial Electronics*, 5 Feb. 2007, doi:10.1109/TIE.2006.888804.

3. Lillicrap, Timothy. "Continuous Control with Deep Reinforcement Learning." *arXiv,* 9 Sept. 2015, arXiv:1509.0297

4. Fortuato, Meie. "Noisy Networks for Exploration." *arXiv*, 30 Jun 2017, arXiv:1706.10295

5. Martius, Georg. "Equation Identification for Extrapolation and Control." MPI for Intelligent Systems, Tubingen Autonomous Learning Group. https://www.is.mpg.de/uploads_file/attachment/attachment/494/Martius_EQL_2018.pdf