USING PHYSICS AS PRIOR IN IMITATION LEARNING

Muhammad Zubair Irshad
School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, GA 30332
mirshad7@gatech.edu

Yoonwoo Kim
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
ywkim0606@gatech.edu

John Seon Keun Yi
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
jyi62@gatech.edu

Spring 2021

ABSTRACT

System identification, a methodology to obtain a dynamic model has been studied extensively in the classical robotics field. Approaches to system identification can be naively divided into two branches: Whitebox models and Blackbox models. In this work, we attempt to learn a model that brings advantages of both branches called Deep Lagrangian Networks through human demonstration to obtain the dynamic model of a humanoid. This model will inherently abide Lagrangian Mechanics and thus obey the laws of physics. We then discuss ideas on how this model can be used in a model-based reinforcement learning setting to generate a physically plausible policy in a sample-efficient way.

1 Introduction

Optimal robotic system control needs accurate dynamic models. The methodology to obtain this dynamic model is called system identification where dynamic models are estimated based on measurements of the system's input and output signals.[1] Approaches to system identification can be naively divided into two branches namely Whitebox models and Blackbox models. Whitebox approaches such as classic regression suffer from inaccuracies via model bias since the assumptions are imposed by human experts. Blackbox approaches on the other hand, uses overparameterized function classes that may require large amounts of trained data to achieve generalization, suffer from risk of over-fitting, and are not interpretable. Since we know that all elements residing in the physical world can be modeled in Lagrangian Mechanics, if we can embed the Lagrangian Mechanics into the system, the estimated dynamic model will obey the laws of physics with fewer training samples. This work has been done by Lutter et al. where Deep Lagrangian Networks was proposed as a deep network structure and learn the equations of motion of a mechanical system with a deep network efficiently while ensuring physical plausibility.[2] In our work, we extended this idea and learn physically plausible dynamic model of a humanoid using human demonstration. Human demonstration was obtained from CMU Motion Capture Database where we used 15 different human motions.

Model-based reinforcement learning for continuous control is a growing research field due to the method's capability of learning to control rapidly in a sample efficient manner, and its ability to integrate specialized domain knowledge into the agent about how the world works. MBRL is known to be particularly effective if the human expert can hand-engineer a dynamic representation using one's knowledge of physics.[3] Learning the physically plausible dynamic model can present an alternative approach to hand-engineering models while preserving the physics prior. Therefore, we argue that MBRL couples well with how we can use the learned physically plausible dynamic model and present how this model can be used to generate a physically consistent policy in a sample efficient manner.

2 Related Works

2.1 System Identification

To learn the parameters of the dynamics model, classical literature uses standard regression or artificial neural networks to fit the forward or inverse dynamics model to the data. Previous works have used linear regression [4, 5], gaussian mixture models [6, 7] to perform regression on the data, support vector regression [8], artificial neural networks i.e. feedforward models [9, 10] and recurrent neural networks [11] to find dynamic parameters or fit forward or backward dynamics to the data.

2.2 Model-free Reinforcement Learning

Model-free reinforcement learning algorithms can learn complex behaviors autonomously when combined with rich function approximators such as, deep neural networks. The model free algorithms are used when the ground-truth model of the environment is not available to the agent. The model free methods has been studied extensively by the AI community and achieved impressive results on tasks ranging from various games to continuous control in robotics. However, model-free algorithms often require a large amount of data to train and arrive at an effective solution and this was crucial downfall when applying them in the real world.

2.3 Model-based Reinforcement Learning

In contrast to Model-free reinforcement learning, Model-based reinforcement learning algorithms obtain a policy in a sample efficient way because given the dynamic model, the agent can plan it's actions. Therefore, MBRL can rapidly learn to continuous control with a limited number of trials. The dynamic model used in MBRL can either be learned or hand-engineered by a human expert. This allows the researchers to integrate specialized domain knowledge into the agent about how the environment works. Once the forward dynamics model is learned, in our work the extended DeLAN model for a humanoid robot, policy optimization algorithm can be used to compute the next action as

$$a_{t+1} \sim \pi(s_t, f) \tag{1}$$

Where the policies are obtained as maximum sum of expected rewards over a predictive horizon.

3 Methodology

3.1 Lagrangian Overview

We learn a model to get physically plausible states in simulation for a dynamical system. We encode physics in our model using a lagrangian [12] dynamics formulation. The lagrangian of a multi-body system is expressed as follows:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}} - \frac{\partial L}{\partial \mathbf{q}} = \mathbf{F} \tag{2}$$

where $L = K - P$ denotes the Lagrangian of the system. Here, K is the kinetic energy and V is the potential energy. The kinetic energy K can be computed as $K = \frac{1}{2}\dot{\mathbf{q}}^T\mathbf{M}(\mathbf{q})\dot{\mathbf{q}}$. $\mathbf{M}$ denotes the the symmetric and positive definite inertia matrix.

Solving the Lagrangian equation yields the following equation of motion for a rigid body manipulator:

$$M(q)\ddot{q} + \dot{q}^\top C(q)\dot{q} + G(q) = \tau \tag{3}$$

Where M is the $n \times n$ Inertia matrix, C is $n \times 1$ Coriolis vector and G is the $n \times 1$ gravity vector.

More generally, the equation of motion can be expressed as follows:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} - \frac{1}{2}\left(\frac{\partial}{\partial \mathbf{q}}\left(\dot{\mathbf{q}}^T\mathbf{M}(\mathbf{q})\dot{\mathbf{q}}\right)\right)^T + \mathbf{G}(\mathbf{q}) = \tau \tag{4}$$

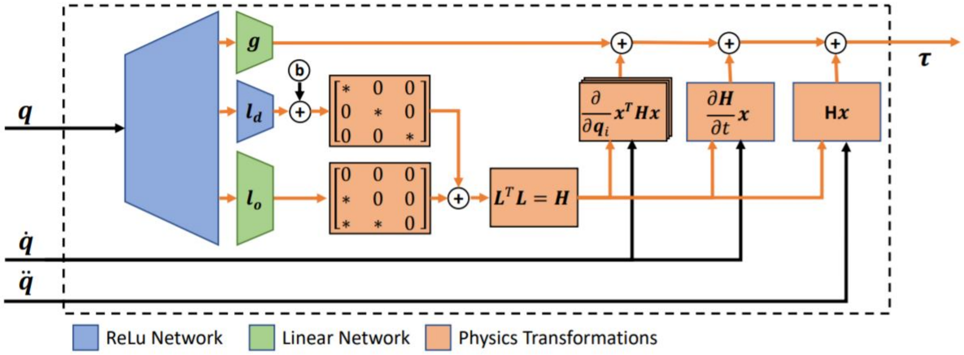
q
q̇
q̈
g
b
l_d
l_o
$\begin{bmatrix} * & 0 & 0 \\ 0 & * & 0 \\ 0 & 0 & * \end{bmatrix}$
$\begin{bmatrix} 0 & 0 & 0 \\ * & 0 & 0 \\ * & * & 0 \end{bmatrix}$
$L^T L = H$
$\frac{\partial}{\partial q_i} x^T H x$
$\frac{\partial H}{\partial t} x$
Hx
τ
ReLu Network
Linear Network
Physics Transformations

Figure 1: Figure Credits: Lutter et al. [2]. Analytical approach to back-propagating a lagrangian neural network

3.2 Deep Lagrangian Networks

Similar to [2], to learn the dynamics, we parametrize each individual block of the langrangian i.e. $\mathbf{M}(\mathbf{q}), \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}), \mathbf{G}(\mathbf{q})$ matrices as a Neural Network. We encode physical consistency in each of the $\mathbf{M}(\mathbf{q}), \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}), \mathbf{G}(\mathbf{q})$ terms in the lagrangian by using a cholesky decomposition for the Mass $(\mathbf{M}(\mathbf{q})$ matrix. Finally, we combine the $\mathbf{M}(\mathbf{q}), \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}), \mathbf{G}(\mathbf{q})$ terms in the lagrangian equation to compute torques in the forward model. We minimize the Mean-Squared Error Loss (MSE) loss between predicted torques and ground truth torques computed using a PD tracking controller.

Formally, we model the dynamics by representing the functions $\mathbf{G}(\mathbf{q})$ and $\mathbf{M}(\mathbf{q})$ as a Multi-Layer Perceptrons. We utilise cholskey decomposition of the mass matrix $\mathbf{M}(\mathbf{q})$ rather than representing $\mathbf{M}(\mathbf{q})$ directly. Ue do this by representing the the lower-triangular matrix $\mathbf{L}(\mathbf{q})$ as a Multi-Layer Perceptron. We write as follows:

$$\mathbf{M}(\mathbf{q}) = \mathbf{L}(\mathbf{q}; \theta)\mathbf{L}(\mathbf{q}; \theta)^T \tag{5}$$

Similar to [2], our learning based model optimization problem to infer the Lagrangian model is to minimize the following inverse dynamics model:

$$f^{-1}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}; \theta, \psi) = \mathbf{L}\mathbf{L}^T\ddot{\mathbf{q}} + \frac{d}{dt}\left(\mathbf{L}\mathbf{L}^T\right)\dot{\mathbf{q}} - \frac{1}{2}\left(\frac{\partial}{\partial \mathbf{q}}\left(\dot{\mathbf{q}}^T\mathbf{L}\mathbf{L}^T\dot{\mathbf{q}}\right)\right)^T + \mathbf{G} \tag{6}$$

where f is the inverse dynamics model and θ and ϕ are neural networks parameters to be learnt.

3.3 Back-propagating gradients

As shown in [2], we cannot solve the deep learning based optimized problem using a gradient based end-to-end approach due to the additional semi-positive definite constraints. Hence we adopt an analytical approach to back propagation as presented by [2] and shown in Figure 1.

3.4 Reinforcement Learning with Learned Dynamics Model

In model-based reinforcement learning, the agent learns a forward dynamics model to approximate a transition function of the environment. [13] The learned physics model obtained from the previous section can be used in place of the forward dynamics model. Since state-action pairs are used for the transition function, we need to decompose the action into $\mathbf{q}, \dot{\mathbf{q}}$ in order to accommodate the physics model.

4 Experiments

In this section we perform a set of experiments to verify if a learned dynamics model of a humanoid performs well in a RL setting. We focus on integrating the learned physics controller with model-based reinforcement learning ... First we present the motion capture simulation data and use the data to train a deep Lagrangian network. Then we outline our plans to use the learned physics model obtained from deep Lagrangian in model-based RL.

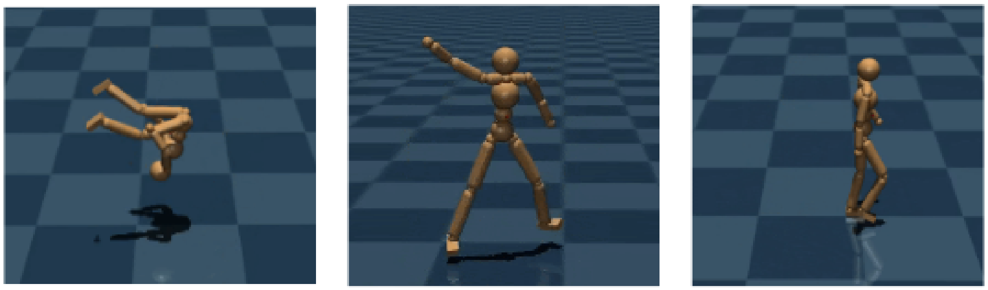

Figure 2: From left to right: Backflip, Dance, Walking motion from the motion capture dataset

4.1 Data Preparation

To prepare data to learn the physics model, we use 3D humanoid motion capture data obtained from the CMU Graphics Lab Motion Capture Database [1]. The 3D humanoid is composed of 28 joints with the position and velocity $(\mathbf{q}, \dot{\mathbf{q}})$ of each joint are provided in the database. From the given $(\mathbf{q}, \dot{\mathbf{q}})$ in each timestep, we calculate acceleration $\ddot{\mathbf{q}}$ and extract torque τ from a tuned PD controller. The position, velocity, acceleration and torque generated from the mocap data is fed into the Lagrangian neural network. 15 different motions such as walking, dance, backflip are used with around 1000 timesteps sampled per motion. The motion capture data is demonstrated though a Mujoco [2] environment. 2 presents captures of three motions from the database played visualized using Mujoco.

4.2 Model Learning Experiments

To demonstrate the effectiveness of the learnt model, we perform an experiment on the MOCAP mujoco dataset for the humnaoid. We learn the inverse dynamics model using our technique and compare it against the end-to-end feedforward baseline. The feedforward baseline is a 3 layer Multi-layer perceptron which maps the input positions, velocities and join accelerations to actions using a hidden intermediate layer. We report training and validaiton loss for both our model and feedforward network.

The results as shown in Figure 3 indicate the faster convergence in train-time as well as better generalization in validation-time of our lagrangian learning approach when compared to the feedforward network. The results also show data insufficiency at higher epochs as the train curve starts to flatten at higher epochs. This might be due to the repetitive motions in the dataset which allows some overfitting of both the models.

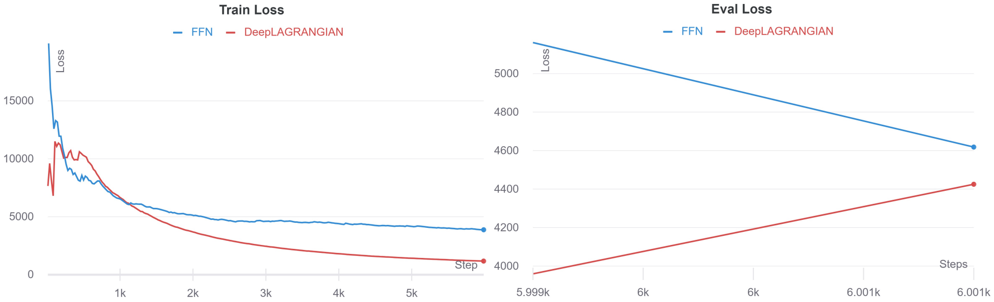

Figure 3: **Model Learning Comparison:** We compare our lagrangian deep learning based model learning with a feedforward network. Our model converges faster with a better generalization in validation dataset.

5 Conclusion and Discussions

In this project, we explore if a physically plausible dynamic model can be learned from demonstration (motion capture) using a Deep Lagrangian Network. Furthermore, we discuss if the learned model can be used in model-based reinforcement learning to generate a good policy in a sample efficient manner. The trained results of the Lagrangian

[1] http://mocap.cs.cmu.edu/

[2] http://www.mujoco.org/index.html

network indicate that it learns and generalizes better to the mocap data compared to the baseline feedforward network. In the future, we plan to integrate the learned physics model in MB-RL and compare results with baseline model based RL methods.

References

[1] Lennart Ljung. System identification. *Wiley encyclopedia of electrical and electronics engineering*, pages 1–19, 1999.

[2] Michael Lutter, Christian Ritter, and Jan Peters. Deep lagrangian networks: Using physics as model prior for deep learning. *arXiv preprint arXiv:1907.04490*, 2019.

[3] Luis Pineda, Brandon Amos, Amy Zhang, Nathan O Lambert, and Roberto Calandra. Mbrl-lib: A modular library for model-based reinforcement learning. *arXiv preprint arXiv:2104.10159*, 2021.

[4] Stefan Schaal, Christopher G Atkeson, and Sethu Vijayakumar. Scalable techniques from nonparametric statistics for real time robot learning. *Applied Intelligence*, 17(1):49–60, 2002.

[5] Masahiko Haruno, Daniel M Wolpert, and Mitsuo Kawato. Mosaic model for sensorimotor learning and control. *Neural computation*, 13(10):2201–2220, 2001.

[6] Sylvain Calinon, Florent D'halluin, Eric L Sauser, Darwin G Caldwell, and Aude G Billard. Learning and reproduction of gestures by imitation. *IEEE Robotics & Automation Magazine*, 17(2):44–54, 2010.

[7] S Mohammad Khansari-Zadeh and Aude Billard. Learning stable nonlinear dynamical systems with gaussian mixture models. *IEEE Transactions on Robotics*, 27(5):943–957, 2011.

[8] Younggeun Choi, Shin-Young Cheong, and Nicolas Schweighofer. Local online support vector regression for learning control. In *2007 International Symposium on Computational Intelligence in Robotics and Automation*, pages 13–18. IEEE, 2007.

[9] Ian Lenz, Ross A Knepper, and Ashutosh Saxena. Deepmpc: Learning deep latent features for model predictive control. In *Robotics: Science and Systems*. Rome, Italy, 2015.

[10] Fernando Díaz Ledezma and Sami Haddadin. First-order-principles-based constructive network topologies: An application to robot inverse dynamics. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 438–445. IEEE, 2017.

[11] Elmar Rueckert, Moritz Nakatenus, Samuele Tosatto, and Jan Peters. Learning inverse dynamics models in o (n) time with lstm networks. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 811–816. IEEE, 2017.

[12] John M Hollerbach. A recursive lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity. *IEEE Transactions on Systems, Man, and Cybernetics*, 10(11):730–736, 1980.

[13] Luis Pineda, Brandon Amos, Amy Zhang, Nathan O. Lambert, and Roberto Calandra. Mbrl-lib: A modular library for model-based reinforcement learning. 2021.