Kinematic Critic: Improving Manipulation Agents by Differentiating through Known Robot Kinematics

Johanna Hansen*[12†] Kyle Kastner*[23] Yuying Huang[14]
Aaron Courville[23‡] David Meger[12†] and Gregory Dudek[12†]

***Abstract*— This paper introduces a method for learning manipulation agents which incorporate robot forward kinematics as a differentiable module for reinforcement learning systems. Forward kinematics as described by Denavit-Hartenberg (DH) parameterization for rigid-body robots is fully differentiable with respect to input joint angles, given fixed link-relative geometric information. Including this differentiable module into the structure of a reinforcement learning agent improves training speed, stability, and overall performance. By incorporating this information only in the critic, the final learned policy used to predict joint actions from image input *does not directly depend* on receiving input joint information, instead learning the necessary behavior implicitly via the kinematics-informed critic. We illustrate this approach by modifying the critic in a modern pixel-based actor-critic baseline to be a Kinematic Critic and ablating across variations that provide similar pose information but without the kinematic bias in the network architecture. Results are given across several manipulation tasks and two robot arms in Robosuite [1]. We additionally demonstrate a simulation-learned policy running on a real Jaco 7DOF robot.**

I. Introduction

This paper incorporates the classic Denavit-Hartenburg (DH) [2] method of parameterizing robot kinematics as a differentiable function for improving reinforcement learning agents on several robotic manipulation tasks. The DH method of joint parameterization was developed in 1955 by Jacques Denavit and Richard Hartenberg [2] and has been a staple tool for defining kinematic functions for rigid-body robots.

Robots learning to solve manipulation tasks must inherently reason about controlling their own body. Models and controllers based on explicit analytic physical parameterization traditionally need detailed information such as manipulator redundancies, kinematic limits, friction, acceleration, and/or inertia, which can be difficult to define exhaustively and measure accurately. When solving control tasks without explicit information about the robot kinematics, such as in model-free reinforcement learning (RL), dynamics and structure prediction are inherently coupled with task-solving in the agent. In this paper, we argue for a medium ground between fully human-defined controllers which require defining parameters such as friction and inertia which are often difficult to accurately estimate, measure, or simulate, and fully learned agents who have no knowledge of the robot they are tasked with controlling. Our approach incorporates the easily defined and (typically) constant factors of robot geometry into the learning pipeline resulting in better overall learned control policies.

Jaco Pick and Place Can in Simulation

Panda Door Open in Simulation

Jaco Reach Ball Real

Fig. 1: Training visual policies with a Denavit-Hartenburg view of robot joint positions enables agents to learn complex visual policies that can operate in the real world. This figure depicts successive frames of our Kinematic Critic performing robotic manipulation tasks. Full videos can be found at https://johannah.github.io/kinematic-critic

Recently introduced automatic differentiation tools such as Pytorch [3] and Jax [4] enable easy incorporation of the DH-defined kinematics function (and its Jacobian) into deep learning models with full gradient propagation. Recent innovations have illustrated the appeal of incorporating differentiable functions [5] into learning systems or building entire environments [6], [7], [8] as a method for improving the learning of complex functions which have differentiable physical simulators. Our approach maintains much of the simplicity of model-free RL while harnessing well-defined robot kinematics as a differentiable structural bias without the overhead of fully differentiable environments.

∗Equal Contribution. [1]McGill University, Montreal, Quebec. [2]Mila, Montreal, Quebec. [3]Université de Montréal, Montreal, Quebec. [4]Stanford University, Stanford, California. †Supported by the National Canadian Robotics Network (NCRN) ‡CIFAR Fellow.

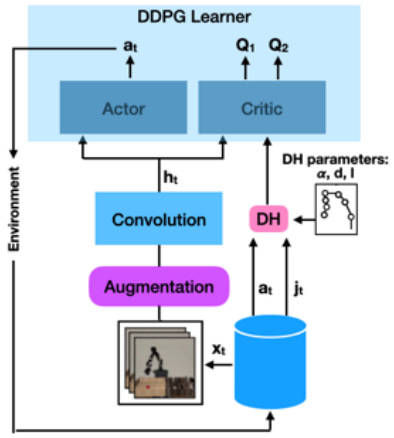

Fig. 2: In the Kinematic Critic architecture, we differentiate through a function performing Denavit-Hartenburg kinematics during training. The DH function is given the constant α, d, and l parameters as well as the joint angle state, j_t, and relative angle action, a_t, which are added together to form θin Eq. 1. Note that the sampling of action a_t admits differentiation with respect to the action distribution predicted by the actor, using the reparameterization trick [9], [10].

The main contributions of this paper are as follows:

- Describe a method to incorporate forward kinematics with the Denavit-Hartenberg (DH) function in an automatic differentiation framework for use in a deep reinforcement learning algorithm.
- Demonstrate notable improvement on a suite of robot manipulation benchmarks in simulation on Jaco and Panda robots over a standard pixel-based actor-critic algorithm.
- Show the importance of differentiating through the kinematic function over architectures that provide the same joint information without the structural bias of kinematics.

II. Background

A. Kinematics

Forward kinematics allows computing a robot's kinematic chain for a particular configuration to find the pose of the end-effector from *joint parameters* and joint angles, where *joint parameters* are known constants that define the geometric relationship between the serial links of a rigid body chain. There are several conventions for assigning joint parameters [11], but in this paper, we only explore the Denavit-Hartenburg [2] method of parameterization. DH is advantageous because it is easily calculable, is often provided by default by robot manufacturers, and is differentiable with respect to input joint angles, given geometric information such as lengths and relative link rotations (see Fig 4).

For each joint, i, in a rigid body, the DH parameters are described by d_i (distance from joint i to the actuator axis $i-1$), θ_i (angle rotation about axis $i-1$), a_i (the distance of joint i along actuator axis $i-1$), and α_i (the angle between actuators of axis i and axis $i-1$).

$$[T] = \prod_{i=1}^{n} {}^{i-1}T_i(\theta_i) \tag{1}$$

In this convention, transformations are serially performed between n links with joint parameters θ_i as specified in Eq. 1. The value of a joint angle, ${}^{i-1}T_i(\theta_i)$, shown in Eq. 2 describes the transformation from link i to the previous link.

$${}^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

In this work, the DH parameters (α, d, and l) are assumed constant throughout training, and not directly optimized via backpropagation or any other means. Joint angles, specified by θ, are optimized by the neural control policy of the reinforcement learning agent.

B. Learned Controllers

The choice of controller to use for robot learning can have a large impact on task difficulty - for instance, a ping-pong robot will need precise control of both end-effector pose and velocity, while the task of box stacking may be simpler to learn with a Cartesian controller [12] where low-level control of joints is handled by a controller specified from expert robot knowledge [13]. In Cartesian (also known as Task) Space, actions are the target position and/or orientation of the end-effector in Euclidean space (x, y, z). These high-level control methods can simplify many tasks but often make assumptions about the operating environment and agent structure in order to perform the complex task of Cartesian space to joint space mapping which may make accounting for nuance such as correcting for a grasped object's weight or avoiding obstacles more difficult.

At the other end of the control spectrum are agents which learn to control robots directly from torque applied to motors [14], [15], [16], [17]. This can make solving tasks requiring longer-term planning challenging, as agents typically need to operate at higher control rates while also learning physical dynamics.

In this paper, we bridge the utility of high and lower-level controllers, utilizing joint-angle control to enables precise control of joints, while biasing this control with a forward kinematics function that provides structural bias of end-effector pose. In a similar manner, JAiLeR [18] details a paradigm for training a joint angle controller which maps from Cartesian space to joint space using model-free RL on proprioceptive state observations. JAiLeR relies on curriculum learning to produce an inverse kinematics controller with similar performance to OSC on goal-conditioned reach tasks, demonstrating the capability to incorporate obstacle avoidance directly into the observation state space of the agent. Unlike JAiLeR, an agent trained with a Kinematic

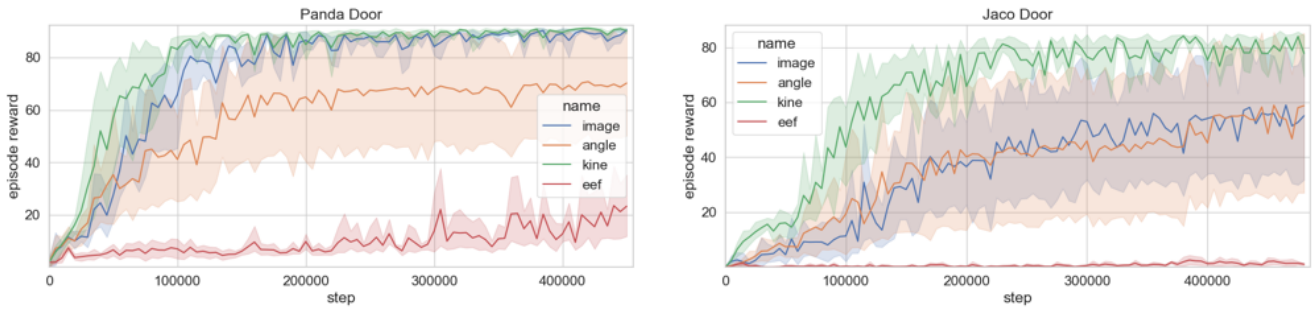

Fig. 3: Evaluation curves depicting the mean (solid line) and the shaded 95 percent confidence interval around the mean, with performance measured over 5 randomly chosen seeds. The Kinematic Critic architecture (green) outperforms other ablations on average, including agents with critics which directly calculate the expected end-effector pose without back-propagating gradient information (red) and agents which are given robot joint angles directly (orange). The stark discrepancy between the green and red reward curves emphasizes the power of allowing gradient propagation.

Critic operates on images rather than joint states during evaluation, requires no special curriculum during training, and is shown to work on other complex tasks in addition to inverse kinematics.

C. Model-Based Control

The Kinematic Critic may be considered a Model-Based Controller as the module computes the forward kinematics for a given agent action though the accuracy of this model is not corrected by any special loss term. Most traditional controllers also use the physical attributes of the robot and/or its environment for operation. In the widely used Operational Space Controller [19], [13], the accuracy of the physical parameterization of the robot, particularly, the mass matrix can vary based on load and configuration, and performance can deteriorate quickly if this estimate is inaccurate [19].

There is a large field of research on developing physics models for use by robot controllers. One can impose varying degrees of prior knowledge about the system into the model, including kinematic equations and factors such as mass, friction, or inertia [20], [21], [22]. Despite its appeal, prescribed knowledge can be difficult or idiosyncratic to define for particular robots, but can potentially provide both generalization and interpretability.

On the other hand, purely data-driven models [17], [23] use function approximators to fit the complex dynamics that govern robot control. Data-driven approaches tend to be limited by the coverage of their dataset, but are often simpler to implement and can excel if the true system characteristics differ from those that might have been prescribed by (often practically limited) model factorizations. Most deployed systems are a blend of prescribed physics with learned models that attempt to overcome the inevitable dynamics errors of our physics assumptions in complex robots. Williams *et al.* [24] demonstrates the power of learning a controller with factorized dynamics models. They incorporate kinematic equations on several robotic systems, including an aggressive driving task. Model-based Action-Gradient-Estimator Policy Optimization (MAGE) [25] is a continuous-control DDPG actor-critic algorithm that explicitly trains the critic to provide action-gradients by backpropagating through a learned dynamics model. OSCAR [26], introduces a data-driven variant of OSC which learns to adapt the physics model online for task-specific and task-agnostic manipulation. For a more exhaustive review of design choices in incorporating learned dynamics models with physics, refer to Lutter *et al.* [20].

Parameter	Value
Controller Type	Joint Position
Control Rate	10 Hz
Impedance Mode	Fixed
Max Action Step	0.15 radians
KP	$(30, 60, 50, 70, 60, 70, 90)$
Damping Ratio	$(0.1, 0.17, 0.2, 0.3, 0.1, 0.1, 0.1)$
Actor Input	3 consecutive RGB frames
Camera	Frontview
Reward	Dense
Expl. Stddev. Schedule	linear$(1.0, 0.1, 500000)$
Environment Uniform Init	Reach: Target 0.1m from EEF Door: $\pm.02m$, ± 0.25rad from origin Lift: Cube $\pm 0.03m$ from origin Can: Can $\pm 0.145m \times \pm 0.195$ region
Robot Joint Init	Gaussian, stddev=0.2

TABLE I: Experiment Hyperparameters

D. Learning Pose with Kinematic Constraints

We study the problem of learning to solve manipulation tasks from images, where the robot must complete a task with

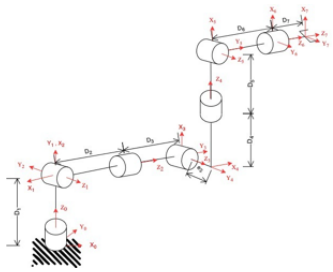

Fig. 4: DH parameters for the Jaco 7DOF Robot.

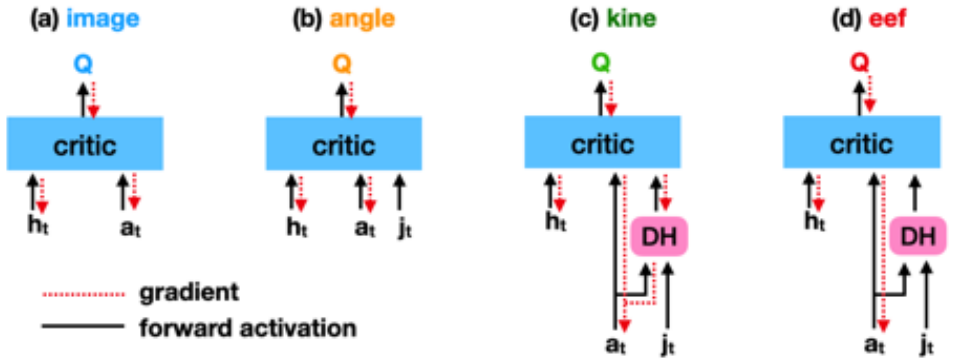

Fig. 5: Diagrams illustrating the 4 critic architectures tested in this paper. The (a) ***image*** variation represents the standard DrQv2 critic setup, the (b) variation shows the critic network with joint ***angle*** directly, (c) is ***kine*** method, which exploits the Denavit-Hartenberg parameterization of forward kinematics to provide a lightweight, differentiable model of the end effector pose to the critic, and finally, in ***eef***, we test the importance of differentiability through the kinematic function by removing the gradients while still giving the critic access to the end effector pose. Critic setups (b), (c), and (d) are a core contribution to this work, and we study the performance of each variant in Section IV.

state information being provided from camera observations. During training, our agent also has access to its own joint angles. When learning robot policies from images, an agent must implicitly learn to map the image and its actions back to its own pose. Learning this action-observation mapping of multi-link articulated objects is inherently difficult because the pose is not only high dimensional but also has structural constraints inherent in the rigid body chain that makes up the robot that are compounding and not always visible in the single-camera view provided. Past work on mixture density estimation using neural networks [27] utilizes a neural network that, given input joint angles and assuming fixed link lengths, predicts the parameters of a mixture distribution over end-effector positions for robot kinematics of a simplified two-link arm. Though this work was not utilized directly for control, this example serves as an introduction to the more general kinematics concepts at play in our work.

The observation problem has been tackled in the computer vision community. Deep Kinematic Pose Regression [28] embeds a differentiable kinematic object model into a neural network for predicting pose from images. Their network predicts the joint motion parameters of an object while learning directly on the joint location loss described by a kinematics chain or kinematics tree. This model is demonstrated on a toy 2D robot and 3D human pose, achieving state-of-the-art results on the Human3.6M dataset. The formulation of the kinematics loss based on pixel input bears similarity to the overall approach featured in our work, however, Deep Kinematic Pose Regression was used in a supervised learning setting with a direct loss on pose, rather than for robotic control.

III. Method

We employ a Deep Deterministic Policy Gradients (DDPG) [29] reinforcement learning agent that utilizes an actor-critic [30] network structure. As in Asymmetric Actor-Critic [31], our approach uses the fact that the actor and critic are two separate networks to give extra information to the critic during training. Our reinforcement learning agent is built on the successful DrQv2 [14] network architecture. DrQv2, which iterated on its predecessor DrQ [32], utilizes image augmentation to achieve sample-efficient high performance on continuous robot controls tasks. DrQv2 employs a DDPG learner with uses n-step returns to estimate TD error. As in DrQ [32], DrQv2 [14], and TD3 [33], the practical implementation of these Q functions uses clipped Double Q-learning, duplicating estimate calculations through independent networks with identical architectures f_1 and f_2, resulting in $Q1_*$ and $Q2_*$. For more details on this formulation, and overall problem setting see DrQv2 [14]. All of our environment hyperparameters match the official implementation of DrQv2, aside from the replay buffer size, which we adjust from 1M to 500k stored image states. To adapt DrQv2 from Deepmind Control Suite [34] Torque Control to Robosuite [1] Joint Position Control, we do not repeat actions, instead, the agent runs its controller requesting relative joint angles at a constant control rate. Please see Tab. I for additional detail.

We test the *image* critic architecture from DrQv2, with *angle*, *kine* (Kinematic Critic), and *eef* ablations as described in Fig. 5. These ablations were chosen to test the influence of the choice of representation of the additional information provided in Kinematic Critic. In the *angle* ablation, current robot joint angles are concatenated with the state, h_t (as encoded by a convolutional network), and actions, though it does not benefit from explicit knowledge of the kinematic structure of the robot. For both *eef* and *kine*, we estimate the expected pose of the end-effector for a given relative action by computing forward kinematics from the current joint angle using the DH method. In *kine*, the Kinematic Critic enables differentiation that can be used by the agent as a gradient path for learning and backpropagation. However, in

Jaco Door Opening Trained with Images and a Kinematic Critic (*kine*)

Jaco Door Opening Trained with Images Only (*image*)

Fig. 6: Representative frames from Kinematic Critic (top row) and Image Only (bottom row) agents solving the Door Opening task. We find that Kinematic Critic agents show higher coordination among joints (especially evident in the Jaco arm), producing policies in which the end effector moves smoothly and efficiently through space.

the *eef* experiments, we *prevent* gradient propagation through the DH function to study the importance of *gradient flow*, while providing access to roughly the same information as the *kine* variant.

$$Q_{im} = f(h_t, a_t) \tag{3}$$
$$Q_{angle} = f(h_t, a_t, j_t) \tag{4}$$
$$Q_{kine} = f(h_t, DH(a_t + j_t)) \tag{5}$$
$$Q_{eef} = f(h_t, DH(sg(a_t) + j_t))) \tag{6}$$

Equation group 6 outlines the core critic calculations used by our tested architectures, as outlined in Figure 5. We denote the parameterized critic architecture as f, sg for "stop gradient", and DH for the Denavit-Hartenburg calculation as described in subsection II-A. h_t is the intermediate hidden activation resulting from a convolutional network over the input image at time t, x_t, $h_t = conv(x_t)$, a_t is the relative action sampled from the actor sub-network, and j_t denotes the absolute joint position, as given by angle encoders on robot joints.

A. Core Contributions

The primary contributions of our method, as compared to the aforementioned background are as follows

- Our work builds directly on a strong actor-critic visual RL baseline with no use of expert traces, imitation learning, or behavior cloning.
- We utilize rewards from the environment and do not formulate pose-specific losses or employ multitask training. We provide end-effector pose information (via a differentiable DH function) through the internal workings of the critic sub-network, thus allowing the overall actor-critic model to decide how best to use this information to maximize overall reward. This means that useful pose-related dynamics (such as smoothness or stability) are learned implicitly, without explicit pose-specific losses or rewards.
- Rather than using the entire feature-based state information in the critic like Asymmetric Actor-Critic [31], we use only joint angles of the robot along with the DH parameterization to improve training. This is advantageous as joint angles are likely to be well-modeled in sim2real transfer for a variety of robot platforms.
- Dynamics learning is relegated to the underlying RL agent, and we do not explicitly factorize dynamics learning using physics knowledge - only kinematic structure, on a per-timestep basis, is used. While kinematic structure in this work is closely related to the adjoint calculations used in the Articulated Body Algorithm (ABA) [35], [21], as well as the kinematic calculations in Deep Kinematic Pose Regression [28], QuaterNet [36], or many other works utilizing kinematic chain or tree calculations, it is not coupled with estimations of velocity, acceleration, mass, inertia, and other physics related quantities. Instead, we utilize a combination of RL and standard joint-position controllers for overall problem dynamics.
- We operate directly from pixels, with no goal conditioning or state information [18] beyond the knowledge of the robot geometry (assumed constant throughout training on a per-task basis, and given to the critic sub-network) and robot joint angles (which are also used by the controller). The actor only consumes images as inputs, as is common in continuous control from pixels work in reinforcement learning [14], [15].

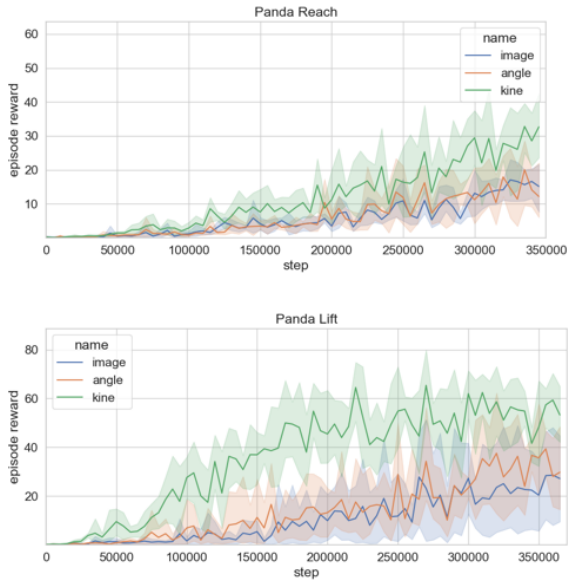

Fig. 7: Reach and Lift tasks on the Panda.

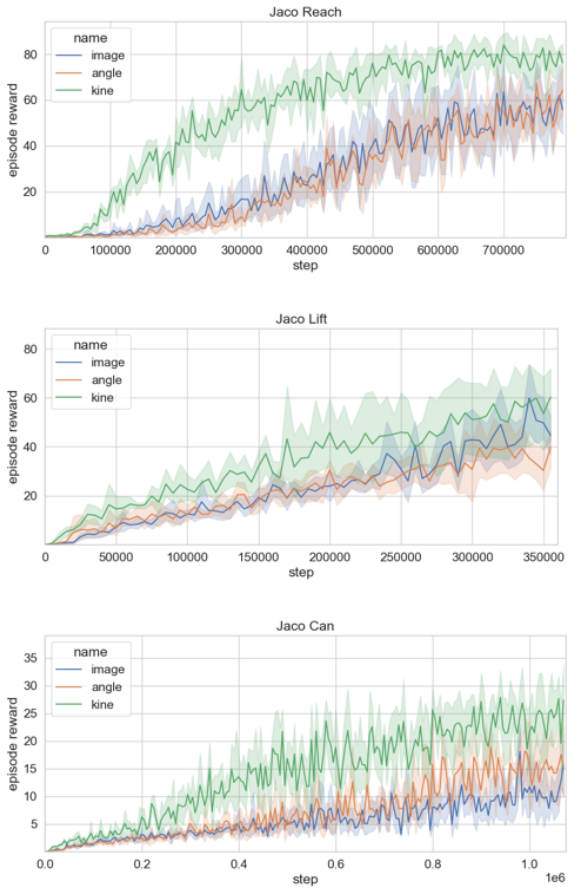

Fig. 8: Reach, Lift, and Can tasks on the Jaco.

IV. Experiments and Discussion

We investigate the impact of adding differentiable kinematics structure into the critic of an image-based reinforcement learning architecture for a set of tasks trained in Robosuite [1], a robotics simulation framework powered by the MuJoCo physics engine [37].

We demonstrate performance on 4 tasks: Door, Reach, Lift, and Can (listed in the order of increasing difficulty) for the Jaco 7DOF manipulator and the Panda 7DOF arm. Object position and robot initial joints are randomized on reset, with range of variability set per task as default in Robosuite. All performance curves are shown depicting the mean (solid line) and the 95 percent shaded confidence interval around the mean over 5 randomly chosen seeds of episodic reward over agent training steps in evaluation.

Our experiments show the differentiability of the DH function is critical to driving effective learning of the network. Simply providing the relevant proprioceptive information to the critic, such as in the *angle* variant (orange), did not seem to help much over the pure *image* variant (blue). This is particularly evident in the reward traces for the Jaco Door opening task shown in Fig I. By propagating information from the DH function inside the critic, through the action space, into the actor and finally through the convolutional encoder networks, we see that the Kinematic Critic agent (green) greatly outperforms the *eef* variant (red) despite both agents receiving the end effector pose as input into the critic. Enabling gradient flow through a kinematics function improves learning speed, stability, and overall performance in every case we tested. This performance increase is particularly notable on tasks where a wide distribution of coordinated poses were required, especially on the Jaco, which was less stable in our simulation.

We also demonstrate the Reach policy on a real Jaco arm (see Fig.1) by iteratively syncing the simulator to the real setting, predicting actions based on simulator frames, and then applying the predicted action back to the real robot. More sophisticated real2sim or sim2real [38] approaches could be integrated symbiotically with the Kinematic Critic to improve performance in this setting, but fall outside the scope of the current publication.

V. Conclusion

In this work, we introduce a method of incorporating differentiable Denavit-Hartenburg (DH) transformations into an actor-critic reinforcement learning algorithm. By training a critic with access to DH while training its actor only on images, we learn vision-based policies for complex manipulation tasks with better performance than variants with access to the same joint state information. Our evaluation shows that the differentiability of the DH transformation in the critic is crucial for effective training. Overall our method improves upon strong actor-critic baselines across several benchmark tasks and solves all tasks. The learned policies are directly transferrable to real-world settings, and we demonstrate this by tasking a real-world robot using simulation learned policies.

REFERENCES

[1] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín, "robosuite: A modular simulation framework and benchmark for robot learning," *CoRR*, vol. abs/2009.12293, 2020. [Online]. Available: https://arxiv.org/abs/2009.12293

[2] R. S. Denavit, Jacques; Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Trans ASME J. Appl. Mech*, vol. 23, pp. 215–221, 1955.

[3] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[4] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018. [Online]. Available: http://github.com/google/jax

[5] D. P. E. R. E. Riba, D. Mishkin and G. Bradski, "Kornia: an open source differentiable computer vision library for pytorch," in *Winter Conference on Applications of Computer Vision*, 2020. [Online]. Available: https://arxiv.org/pdf/1910.02190.pdf

[6] Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand, "Difftaichi: Differentiable programming for physical simulation," *ICLR*, 2020.

[7] K. M. Jatavallabhula, M. Macklin, F. Golemo, V. Voleti, L. Petrini, M. Weiss, B. Considine, J. Parent-Levesque, K. Xie, K. Erleben, L. Paull, F. Shkurti, D. Nowrouzezahrai, and S. Fidler, "gradsim: iulation for system identification and visuomotor control," 2021.

[8] P. Gradu, J. Hallman, D. Suo, A. Yu, N. Agarwal, U. Ghai, K. Singh, C. Zhang, A. Majumdar, and E. Hazan, "Deluca - A differentiable control library: Environments, methods, and benchmarking," *CoRR*, vol. abs/2102.09968, 2021.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[10] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, pp. 229–256, 1992. [Online]. Available: https://doi.org/10.1007/BF00992696

[11] "A comparison between the denavit hartenberg and the screw-based methods used in kinematic modeling of robot manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 4, pp. 723–728, 2011, conference papers of Flexible Automation and Intelligent Manufacturing.

[12] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1010–1017.

[13] O. Khatib, "Inertial properties in robotic manipulation: An object-level framework," *The International Journal of Robotics Research*, vol. 14, no. 1, pp. 19–36, 1995.

[14] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Mastering visual continuous control: Improved data-augmented reinforcement learning," *arXiv preprint arXiv:2107.09645*, 2021.

[15] A. Srinivas, M. Laskin, and P. Abbeel, "Curl: Contrastive unsupervised representations for reinforcement learning," *arXiv preprint arXiv:2004.04136*, 2020.

[16] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv preprint arXiv:1912.01603*, 2019.

[17] M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*. Citeseer, 2011, pp. 465–472.

[18] V. Kumar, D. Hoeller, B. Sundaralingam, J. Tremblay, and S. Birchfield, "Joint space control via deep reinforcement learning," *International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[19] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: A theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, 2008.

[20] M. Lutter, L. Hasenclever, A. Byravan, G. Dulac-Arnold, P. Trochim, N. Heess, J. Merel, and Y. Tassa, "Learning dynamics models for model predictive agents," *arXiv preprint arXiv:2109.14311*, 2021.

[21] M. Lutter, J. Silberbauer, J. Watson, and J. Peters, "A differentiable newton-euler algorithm for real-world robotics," 2021.

[22] S. East, M. Gallieri, J. Masci, J. Koutník, and M. Cannon, "Infinite-horizon differentiable model predictive control," *arXiv preprint arXiv:2001.02244*, 2020.

[23] J. C. G. Higuera, D. Meger, and G. Dudek, "Synthesizing neural network controllers with probabilistic model-based reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2538–2544.

[24] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1714–1721.

[25] P. D'Oro and W. Jaśkowski, "How to learn a useful critic? model-based action-gradient-estimator policy optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 313–324, 2020.

[26] J. Wong, V. Makoviychuk, A. Anandkumar, and Y. Zhu, "Oscar: Data-driven operational space control for adaptive and robust robot manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[27] C. M. Bishop, "Mixture density networks," Tech. Rep., 1994.

[28] X. Zhou, X. Sun, W. Zhang, S. Liang, and Y. Wei, "Deep kinematic pose regression," *CoRR*, vol. abs/1609.05317, 2016. [Online]. Available: http://arxiv.org/abs/1609.05317

[29] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2019.

[30] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," in *Advances in Neural Information Processing Systems*, S. Solla, T. Leen, and K. Müller, Eds. MIT Press.

[31] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," 2017.

[32] D. Yarats, I. Kostrikov, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=GY6-6sTvGaf

[33] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[34] Y. Tassa, S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, and N. Heess, "dmcontrol: Software and tasks for continuous control," 2020.

[35] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2014.

[36] D. Pavllo, D. Grangier, and M. Auli, "Quaternet: A quaternion-based recurrent model for human motion," in *British Machine Vision Conference (BMVC)*, 2018.

[37] E. Todorov, T. Erez, and Y. Tassa.

[38] M. Mozifian, A. Zhang, J. Pineau, and D. Meger, "Intervention design for effective sim2real transfer," 2020.