Adversarially Regularized Policy Learning Guided by Trajectory Optimization

Zhigen Zhao*, Simiao Zuo†, Tuo Zhao†‡ and Ye Zhao*‡
*School of Mechanical Engineering, Georgia Institute of Technology
†School of Industrial and Systems Engineering, Georgia Institute of Technology
‡Co-corresponding Authors
§ {zhigen.zhao, simiaozuo, tourzhao, yzhao301} @gatech.edu

Abstract—**Recent advancement in combining trajectory optimization with function approximation (especially neural networks) shows promise in learning complex control policies for diverse tasks in robot systems. Despite their great flexibility, the large neural networks for parameterizing control policies impose significant challenges. The learned neural control policies are often overcomplex and nonsmooth, which is inconsistent with the fact that optimal control policies are smooth with respect to state for most robotic systems. Therefore, they often yield poor generalization performance in practice. To address this issue, we propose adVErsarially Regularized pOlicy learNIng guided by trajeCtory optimizAtion (VERONICA) for learning smooth control policies. Specifically, our proposed approach controls the smoothness (local Lipschitz continuity) of the neural control policies by stabilizing the output control with respect to the worst-case perturbation to the input state. Our experiments on robot manipulation show that our proposed approach not only improves the sample efficiency of neural policy learning but also enhances the robustness of the policy against various types of disturbances, including sensor noise, environmental uncertainty, and model mismatch.**

I. Introduction

Robust and generalizable motion planning enables robotic systems to handle various uncertainties and accomplishes diverse tasks. However, learning a dynamically consistent neural control policy (i.e., a neural-network control policy) and executing it reliably remain challenging. First, the function approximators used to model the policy can be highly complex and non-smooth, causing poor generalization performance. Second, the dynamics models involved often have some mismatch between the physical robot and the environment, leading for the need to learn a robust policy.

The authors of [7, 5] take advantage of both trajectory optimization (TO) and policy search by training a robot control policy supervised by optimized trajectory samples, and meanwhile adapting TO to the learned policy. The work in [7] observes that the derivatives of a neural control policy can behave irregularly even when the policy matches the optimal trajectory baseline. This is because neural networks have high complexity and flexibility, which makes them highly non-smooth — a small change in the networks' input can cause a large variation in the output.

To alleviate these issues, we propose a new approach: ad**VE**rsarially **R**egularized p**O**licy lear**NI**ng guided by traje**C**tory optimiz**A**tion (VERONICA). Specifically, our approach improves the local Lipschitz continuity of the neural control policy via adversarial regularization. Our motivation for promoting smoothness in policy stems from the fact that many robotics systems are naturally governed by differential equations with high-order continuity. Namely, similar states should lead to similar optimal controls. We show that VERONICA produces a smooth neural control policy, which improves generalization performance for inputs not seen during training.

We further observe that besides promoting policy smoothness, adversarial regularization improves the robustness of the policy against modeling errors and perturbations in the environment. We verify that the VERONICA framework produces stable robot behaviors under sensor noise, environmental uncertainty, and model mismatch.

Conventionally, adversarial regularization involves a min-max game, which is solved by alternating gradient descent-ascent. During training, neither of the players can be advantageous, such that the generated perturbations can be overstrong and hinder model generalization. To resolve this issue, we employ Stackelberg adversarial regularization (SAR) [12], which formulates adversarial regularization as a Stackelberg game. In SAR, the policy (i.e., the leader) has a higher priority than the perturbation (i.e., the follower). The leader procures its advantage by considering how the follower will respond after observing the leader's decision, such that the leader anticipates the predicted move of the follower when optimizing its strategy. We note that prioritizing the policy optimization is reasonable and beneficial because we target the performance of the learned policy, instead of the adversary.

Our contributions are: I) We propose VERONICA, an adversarial regularization method for learning smooth neural control policies guided by TO. This improves the generalization performance of the learned policy; II) We show that the learned policy achieves better robustness under disturbances such as sensor noise, environmental uncertainty, and model mismatch; III) We reformulate adversarial regularization as a Stackelberg game, which further improves generalization and robustness of the policy compared with the conventional formulation.

II. Method

We introduce VERONICA, our proposed adversarially regularized approach which combines the strength of policy learning and trajectory optimization. First, we define an adversarial regularizer and explain how it improves smoothness and robustness of neural control policies; Second, we describe an ADMM-based algorithm that solves the full joint optimization problem; Third, we develop an extension to our proposed adversarial regularization approach — Stackelberg adversarial regularization. We consider the neural control policy learning process guided by N optimal trajectories $\{\mathbf{X}, \mathbf{U}\} = \{\mathbf{X}_i, \mathbf{U}_i \mid i = 1, \cdots, N\}$, and each optimal trajectory $\{\mathbf{X}_i, \mathbf{U}_i\}$ consists of T state-control pairs $\{\mathbf{x}_i^t \in \mathbb{R}^{d_x}, \mathbf{u}_i^t \in \mathbb{R}^{d_u} \mid t = 1, \cdots, T\}$, where $\mathbf{x}_i^t$ and $\mathbf{u}_i^t$ denote the robot state and the control, respectively. In this study, the robot state corresponds to the joint positions, velocities and task parameters such as goal configurations, while the control corresponds to the joint torque. Moreover, let $\pi(\cdot|\mathbf{W})$ denotes the neural control policy, where $\mathbf{W}$ denotes the associated parameters.

A. Adversarial Regularization for Neural Control Policy

To promote smoothness of the neural control policy, we consider the following adversarial discrepancy measure:

$$\begin{aligned} r_{\text{adv}}(\mathbf{x}, \mathbf{W}) &= \max_{\|\delta\| \le \epsilon} r(\mathbf{x}, \mathbf{W}, \boldsymbol{\delta}) \\ &= \max_{\|\delta\| \le \epsilon} \|\pi(\mathbf{x}|\mathbf{W}) - \pi(\mathbf{x} + \boldsymbol{\delta}|\mathbf{W})\|^2, \end{aligned}$$

where $\|\cdot\|$ denotes the ℓ_2 norm, $\boldsymbol{\delta} \in \mathbb{R}^{d_x}$ is the adversarial perturbation injected to the state vector $\mathbf{x}$, and $\epsilon > 0$ is the perturbation strength. Such an adversarial discrepancy measure $r_{\text{adv}}(\mathbf{x}, \mathbf{W})$ essentially computes the maximal deviation of the neural control policy output at state $\mathbf{x}$ given an input perturbation $\boldsymbol{\delta}$ whose ℓ_2 norm is bounded by ϵ.

We then apply the adversarial discrepancy measure to control the smoothness of the neural control policy. Specifically, we solve the following joint optimization problem:

$$\min_{\mathbf{X}, \mathbf{U}, \mathbf{W}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{X}_i, \mathbf{U}_i) + \mathcal{Q}_{\text{BC}}(\mathbf{X}, \mathbf{U}, \mathbf{W}) + \alpha \mathcal{R}_{\text{adv}}(\mathbf{X}, \mathbf{W}), \tag{1}$$

where $\mathcal{L}(\mathbf{X}_i, \mathbf{U}_i)$ denotes the loss function of the trajectory optimization (TO) for the i^{th} trajectory, $\mathcal{Q}_{\text{BC}}(\mathbf{X}, \mathbf{U}, \mathbf{W})$ denotes the loss function for policy learning:

$$\mathcal{Q}_{\text{BC}}(\mathbf{X}, \mathbf{U}, \mathbf{W}) = \frac{1}{N} \sum_{i,t} ||\pi(\mathbf{x}_i^t|\mathbf{W}) - \mathbf{u}_i^t||^2,$$

$\mathcal{R}_{\text{adv}}(\mathbf{X}, \mathbf{W})$ is the adversarial regularizer for controlling the smoothness of the policy:

$$\begin{aligned} \mathcal{R}_{\text{adv}}(\mathbf{X}, \mathbf{W}) &= \frac{1}{N} \sum_{i,t} r_{\text{adv}}(\mathbf{x}_i^t, \mathbf{W}) \\ &= \frac{1}{N} \sum_{i,t} \max_{\|\boldsymbol{\delta}_i^t\| \le \epsilon} \|\pi(\mathbf{x}_i^t|\mathbf{W}) - \pi(\mathbf{x}_i^t + \boldsymbol{\delta}_i^t|\mathbf{W})\|^2, \end{aligned}$$

and α is the regularization coefficient weighting between the $\mathcal{Q}_{\text{BC}}(\mathbf{X}, \mathbf{U}, \mathbf{W})$ and $\mathcal{R}_{\text{adv}}$.

Solving the optimization problem in Eq. (1) learns a neural control policy that not only minimizes the TO loss and the behavior cloning loss, but also encourages the adversarial discrepancy measure of the policy to be small at every state of the optimal trajectories.

(I) Adversarial Regularization Improves Generalization: Existing methods usually train neural control policies by only minimizing the trajectory optimization loss and behavior cloning loss. Due to the high capacity of deep neural networks, the learned neural control policies are often over-complex and highly non-smooth. This is inconsistent with observations that many optimal control policies for robots are smooth, which requires a small perturbation to the state vector $\mathbf{x}$ to only yield a small change to the policy output. Such a property can improve generalization of the learned policy.

VERONICA naturally promotes the desired smoothness by imposing a high penalty when the adversarial perturbation $\boldsymbol{\delta}$ yields a large deviation to the policy output. More precisely, $r_{\text{adv}}(\mathbf{x}, \mathbf{W})$ essentially upper bounds the deviation of the policy output due to the adversarial perturbation $\boldsymbol{\delta}$ with respect to the state $\mathbf{x}$, and therefore can be viewed as a measure of the local Lipschitz constant within a small neighborhood of $\mathbf{x}$, i.e., $C_{\mathbf{x}} = \sup_{\|\boldsymbol{\delta}\| \le \epsilon} \frac{\|\pi(\mathbf{x}|\mathbf{W}) - \pi(\mathbf{x}+\boldsymbol{\delta}|\mathbf{W})\|}{\|\boldsymbol{\delta}\|}$. Accordingly, our proposed adversarial regularizer penalizes the average discrepancy measures of the neural control policy at all trajectory points, which enforces its local Lipschitz continuity.

(II) Adversarial Regularization Gains Robustness: Robot systems measure their states from sensors, which are prone to stochastic or systematic sensor errors. VERONICA naturally gains robustness against such disturbances. Specifically, the adversarial perturbation in VERONICA can be viewed as a proxy to the errors. Therefore, our approach does not require prior knowledge of them. In comparison, existing methods for handling such errors usually assume specific forms, e.g., independent Gaussian noise, which can be restrictive in practice.

Moreover, as suggested in [1], the Lipschitz continuity is essential to robustness, especially for control and reinforcement learning problems. This is because for policies without the Lipschitz continuity property, a small error in sensor measurement or state transition potentially leads to a drastic change to the policy output. Due to the dynamic nature of the control problem, it will further yield significant error compounding during policy roll-out. As the VERONICA approach can effectively control the local Lipschitz continuity of the neural control policy, such an issue can be mitigated.

B. Combined Trajectory Optimization and Adversarially Regularized Policy Learning

We apply ADMM [10, 11] to solve the optimization problem in Eq. (1). Specifically, we reparameterize Eq. (1) into a decomposable form by introducing two auxiliary sets of state and control variables: $(\mathbf{X}^{\text{TO}}, \mathbf{U}^{\text{TO}})$ represents the trajectory samples generated by trajectory optimization (TO), and $(\mathbf{X}^{\text{PL}}, \mathbf{U}^{\text{PL}})$ are copies of $(\mathbf{X}^{\text{TO}}, \mathbf{U}^{\text{TO}})$ for policy learning. Accordingly, the optimization problem in Eq. (1) is reformulated as:

$$\begin{aligned} \min_{\mathbf{X}^{\text{TO,PL}}, \mathbf{U}^{\text{TO,PL}}, \mathbf{W}} & \sum_{i=1}^{N} \mathcal{L}(\mathbf{X}_i^{\text{TO}}, \mathbf{U}_i^{\text{TO}}) \\ & + \mathcal{Q}_{\text{BC}}(\mathbf{X}^{\text{PL}}, \mathbf{U}^{\text{PL}}, \mathbf{W}) + \alpha \mathcal{R}_{\text{adv}}(\mathbf{X}^{\text{PL}}, \mathbf{W}) \\ \text{s.t.} \ & \mathbf{X}^{\text{TO}} = \mathbf{X}^{\text{PL}}, \mathbf{U}^{\text{TO}} = \mathbf{U}^{\text{PL}}. \end{aligned} \tag{2}$$

ADMM splits the above optimization problem into N individual TO problems and a policy learning problem to be solved in an iterative manner. Note that the ADMM update for policy learning at the p^{th} iteration, as seen in Eq. (3), is a min-max optimization problem, which is solved via alternating gradient descent/ascent:

$$\begin{aligned}\mathbf{W}^{p+1} = \arg\min_{\mathbf{W}} \; & \mathcal{Q}_{\text{BC}}(\mathbf{X}^{\text{PL},p}, \mathbf{U}^{\text{PL},p}, \mathbf{W}) \\ & + \mathcal{R}_{\text{adv}}(\mathbf{X}^{\text{PL},p}, \mathbf{W}). \end{aligned} \tag{3}$$

C. Stackelberg Adversarial Regularization

One major limitation of the adversarial regularizer in Eq. (3) is that it solves a min-max-game-based optimization, where neither of the players can be advantageous. This is problematic because the adversarial player may generate over-strong perturbations that hinder generalization. To mitigate this issue, we employ Stackelberg adversarial regularization [12] to solve the policy update in Eq. (3) through a Stackelberg game formulation. In a Stackelberg game, there are two players, a leader (the policy) and a follower (the perturbations). The leader acknowledges the strategy of the follower, such that it is always in an advantageous position. This effectively eliminates the over-strong perturbations.

To simplify the notation, we omit the indices on the trajectory sample points $\mathbf{x}$. We solve

$$\min_{\mathbf{W}} \mathcal{Q}_{\text{SAR}}(\mathbf{W}) = \mathcal{Q}_{\text{BC}}(\mathbf{X}, \mathbf{U}, \mathbf{W}) + \frac{\alpha}{N} \sum r(\mathbf{x}, \mathbf{W}, \boldsymbol{\delta}^K), \tag{4}$$

$$\text{s.t. } \boldsymbol{\delta}^K(\mathbf{W}) = U^K \circ U^{K-1} \circ \cdots \circ U^1(\boldsymbol{\delta}^0).$$

The policy parameter $\mathbf{W}$ in Eq. (4) is the leader, and the perturbation $\boldsymbol{\delta}(\mathbf{W})$ is the follower. Here, $\circ$ denotes operator composition, i.e., $f(\cdot) \circ g(\cdot) = f(g(\cdot))$. Each U^k for $k = 1, \cdots, K$ represents the k^{th} step update operator for the follower's strategy. The operators are defined by pre-selected optimization algorithms such as stochastic gradient descent (SGD) or Adam [4].

In Stackelberg adversarial training, the leader acknowledges the strategy of the follower by treating the perturbations (the follower) as a function of the policy parameters (the leader). Correspondingly, we solve for the policy parameters using gradient descent, where the Stackelberg gradient is

$$\begin{aligned}\frac{\mathrm{d}\mathcal{Q}_{\text{SAR}}(\mathbf{W})}{\mathrm{d}\mathbf{W}} = & \underbrace{\frac{\mathrm{d}\mathcal{Q}_{\text{BC}}(\mathbf{X}, \mathbf{U}, \mathbf{W})}{\mathrm{d}\mathbf{W}} + \alpha \frac{\partial r(\mathbf{x}, \mathbf{W}, \boldsymbol{\delta}^K)}{\partial \mathbf{W}}}_{\text{leader}} \\ & + \underbrace{\alpha \frac{\partial r(\mathbf{x}, \mathbf{W}, \boldsymbol{\delta}^K)}{\partial \boldsymbol{\delta}^K} \frac{\mathrm{d}\boldsymbol{\delta}^K}{\mathrm{d}\mathbf{W}}}_{\text{leader-follower interaction}}. \end{aligned} \tag{5}$$

In comparison, the conventional adversarial regularization in Eq. (3) uses only the leader term and does not consider the leader-follower interaction.

This Stackelberg gradient can be efficiently computed using deep learning libraries, such as *PyTorch* [9]. Please refer to [12] for more details.

III. Results & Future Works

The VERONICA framework is evaluated on Kuka arm manipulation tasks in simulation. The simulation environment is implemented in *PyBullet* [2]. We use *Crocoddyl* [6] as the TO solver. The adversarially regularized policy learning algorithm described in Sec.II-A is implemented using *PyTorch* [9] and *Higher* [3]. We compare generalization and robustness of policies trained with Gaussian perturbations, conventional adversarial regularization (VERONICA-AR), and SAR (VERONICA-SAR). As seen in Figure 1, the adversarially regularized policies consistently outperform the policy trained with no perturbation or Gaussian perturbation. Furthermore, VERONICA-SAR leads to stable and near-optimal robot motions across all attempts, even under a strong sensor noise, while a small percentage of policy roll-outs for VERONICA-AR results in unstable robot motion. This confirms our hypothesis that VERONICA-SAR helps enhance numerical stability comparing to VERONICA-AR.

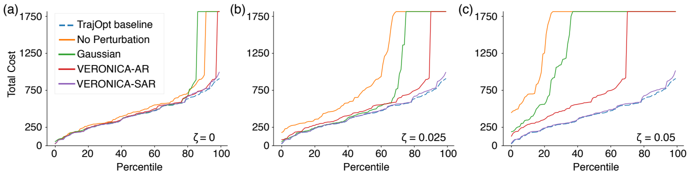

Fig. 1. Cost percentile plot for 3-DOF arm reaching task with 100 different initializations and under disturbances on sensor measurement. Disturbances are drawn from a uniform distribution bounded by ζ. Policies trained with no perturbation, Gaussian perturbation, VERONICA-AR, and VERONICA-SAR are compared against an undisturbed TO baseline. The plot is capped at 2 times the maximum baseline cost. A cost curve that exceeds the plotting cap indicates that a percentage of policy roll-outs lead to unstable robot motion.

We investigate how the performance of VERONICA-SAR scales to higher state and control dimensions by evaluating the task errors of manipulator reaching tasks for 3, 5, and 7-DOF Kuka arms. Table I shows the average task errors over 100 initializations. The task error increases with the dimensionality of the problem, but not significantly. Note that the 5 and 7-DOF experiments involve manipulation in the 3-D space, which lead to much higher problem complexity than the planar 3-DOF Kuka arm configuration, and require larger neural control policies.

TABLE I
Task Errors for M-DOF Manipulator (*Unit: m*)

$M = 3$	$M = 5$	$M = 7$
6.39e-2	1.23e-1	1.32e-1

Our future work will (i) evaluate the performance of VERONICA in the presence of more types of perturbations and uncertainties, such as varying link moment of inertia and kinematic parameters; (ii) extend VERONICA to solve more complex manipulation problems involving physical contact and enhance robustness to contact uncertainties. We will employ a smoothed contact solver similar to the one in [8] to circumvent the discontinuity due to contact phenomena and leverage the smoothness merit induced by the adversarial regularization.

Acknowledgments

This work was supported by the *Ford Motor Company*.

References

[1] Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 264–273. PMLR, 2018.

[2] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

[3] Edward Grefenstette, Brandon Amos, Denis Yarats, Phu Mon Htut, Artem Molchanov, Franziska Meier, Douwe Kiela, Kyunghyun Cho, and Soumith Chintala. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*, 2019.

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Sergey Levine and Vladlen Koltun. Guided policy search. In *International conference on machine learning*, pages 1–9. PMLR, 2013.

[6] Carlos Mastalli, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Justin Carpentier, Ludovic Righetti, Sethu Vijayakumar, and Nicolas Mansard. Crocoddyl: An Efficient and Versatile Framework for Multi-Contact Optimal Control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[7] Igor Mordatch and Emo Todorov. Combining the benefits of function approximation and trajectory optimization. In *Robotics: Science and Systems*, volume 4, 2014.

[8] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG)*, 31(4):1–8, 2012.

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep\-learning-library.pdf.

[10] Zhigen Zhao, Ziyi Zhou, Michael Park, and Ye Zhao. Sydebo: Symbolic-decision-embedded bilevel optimization for long-horizon manipulation in dynamic environments. *arXiv preprint arXiv:2010.11078*, 2020.

[11] Ziyi Zhou and Ye Zhao. Accelerated admm based trajectory optimization for legged locomotion with coupled rigid body dynamics. In *American Control Conference*, pages 5082–5089, 2020.

[12] Simiao Zuo, Chen Liang, Haoming Jiang, Xiaodong Liu, Pengcheng He, Jianfeng Gao, Weizhu Chen, and Tuo Zhao. Adversarial training as stackelberg game: An unrolled optimization approach. *arXiv preprint arXiv:2104.04886*, 2021.