Open-Sourcing Generative Models for Data-driven Robot Simulations

Eran Bamani
School of Mechanical Engineering
Tel-Aviv University
Tel-Aviv, Israel 6997801
eranbamani@mail.tau.ac.il

Osher Azulay
School of Mechanical Engineering
Tel-Aviv University
Tel-Aviv, Israel 6997801
osherazulay@mail.tau.ac.il

Anton Gurevich
School of Mechanical Engineering
Tel-Aviv University
Tel-Aviv, Israel 6997801
antong@mail.tau.ac.il

Avishai Sintov
School of Mechanical Engineering
Tel-Aviv University
Tel-Aviv, Israel 6997801
sintov1@tauex.tau.ac.il

Abstract

Open-source robot hardware has become popular in recent years due easy and low-cost fabrication with 3D printing. Applying reinforcement learning algorithms to these robots, however, require the collection of a large amount of data during robot execution. The process is time consuming and can damage the robot. In addition, data collected for one robot may not be applicable for a similar one due to inherent uncertainties (e.g., friction, compliance, etc.) in the fabrication process. Therefore, we propose to disseminate a generative model rather than actual recorded data. We propose to use a limited amount of real data on a robot to train a Generative Adversarial Network (GAN). We show on two robotic systems that training a regression model using generated synthetic data provides transition accuracy at least as good as real data. Such model could be open-sourced along with the hardware to provide easy and rapid access to research platforms.

1 Introduction

Training Reinforcement Learning policies on real robots is a tedious and time consuming task. The robot must work for a very long time which may cause damage and wear. Therefore, the resulting policies are usually limited to simple tasks while the hardware can do much more. Training in simulation, on the other hand, is a compelling solution where the data is acquired at a lower cost [20]. Simulation-based learning provides a cost-effective way to collect data through interaction with the environment. Such approach, for instance, was used for obtaining control for an autonomous vehicle using a simulation with synthetic images [14]. A similar approach was used for autonomous soil excavation [1, 13]. However, simulations rarely capture reality and the trained policies are usually poorly transferred [15]. This problem is even worse for open-source hardware such as for underactuated robotic hands [12] and mobile robots [19]. Such hardware is usually 3D printed which impose many fabrication uncertainties in, for example, friction, size, mass and compliance. Therefore, hand-crafting models for these systems is a challenging problem leading to the lack of a good simulator.

Learning a policy solely from a simulation and deploying it to the real world is considered a hard challenge. This problem is commonly referred as the *reality gap* or *sim-to-real* (simulation to reality). A common approach to bridge the reality gap is to collect data from the real robot in order to generate

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

a data-driven simulation. The training data is used to learn a *transition model* which maps a current state of the robot and a desired action to the next state. This is commonly done using a supervised learning method such as an Artificial Neural Network (ANN) or Gaussian Processes [18]. This approach yet requires a large amount of training data, may be time consuming, can damage the robot and pose danger to its surrounding.

In this work, we explore the possibility of investing the recorded data in a generative model rather than directly to a regression model. In recent years, attention has been put in developing ANN models that can capture the complex distribution of some data and generate artificial data from the same distribution [17]. Some of the approaches include Variational Auto-Encoder (VAE) [10], Generative Adversarial Network (GAN) [9], Deep Convolutional GAN (DC-GAN) [16], a fully connected and convolutional GAN (FCC-GAN) [2] and Cycle-GAN [23]. We focus in our work on GAN which is a powerful method to learn complex data distributions. GAN is mostly used for image processing [21, 3] and visual perception [5] while also having other applications such as in health care [6, 7, 22] and motion planning [11].

We propose to train a generative model that can provide an infinite amount of synthetic transition data. The generative model would be trained over a limited amount of real data recorded from one robot. Hence, the required effort to collect data is reduced. Furthermore, GAN inherently augments the training data and enables transferring a trained transition model to similar robots with some uncertainties. With all that, open-source hardware (e.g., 3D printed underactuated hands [12]) can be accompanied with an already trained GAN. Therefore, open-source dissemination of the generative model rather than data would be easier and provide flexibility for the prospective user. The user will have a deployment-ready data generator which can cope with fabrication uncertainties of open-source hardware. The generator can immediately provide synthetic data for a custom simulator.

2 Method

2.1 Problem Formulation

Let $\mathbf{x} \in \mathcal{C}$ be the state of robot A where $\mathcal{C} \subset \mathbb{R}^n$ is the configuration space. Further, $\mathbf{a} \in \mathcal{U}$ is an action exerted on the robot where $\mathcal{U} \subset \mathbb{R}^m$ is the action space. Assuming a Markov Decision Process (MDP), the motion of the robot is governed by the function $f : \mathcal{C} \times \mathcal{U} \to \mathcal{C}$ such that, given the current state $\mathbf{x}_t$ and action $\mathbf{a}_t$, the next state is given by $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{a}_t)$. However, an analytical formulation of transition model f is commonly not available or inadequate due to fabrication inaccuracies and inherent uncertainties in the environment. Therefore, we aim to learn a transition model $\tilde{f}$ trained over data from the robot. Nevertheless, a model f_i particularly tuned for robot A_i will, most likely, yield erroneous predictions for robot A_j. Hence, we aim to learn a transition model $\tilde{f}$ trained over some data, and can be independently applied to a similar robot.

2.2 Data collection

Training data is collected by driving the robot in the state space with random actions. During motion, ground-truth data of trajectories is provided. Thus, the resulting data is a set of observed states and actions $\mathcal{P} = \{(\mathbf{x}_1, \mathbf{a}_1), \ldots, (\mathbf{x}_N, \mathbf{a}_N)\}$. The trajectories in $\mathcal{P}$ are processed to a set of inputs $(\mathbf{x}_i, \mathbf{a}_i)$ and corresponding labels of the next state $\mathbf{x}_{i+1}$. Hence, the training data for transition model $\mathbf{x}_{i+1} = \tilde{f}(\mathbf{x}_i, \mathbf{a}_i)$ is of the form $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{a}_i), (\mathbf{x}_{i+1})\}_{i=1}^{N-1}$. Note that in some systems, directly learning the next state can be difficult when the sampling frequency is high and consecutive states are too similar. Therefore, it is common to learn the change from state $\mathbf{x}_t$ given an action $\mathbf{a}_t$ over the time step Δt. Hence, the training dataset will be of the form $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{a}_i), (\Delta\mathbf{x}_{i+1})\}_{i=1}^{N-1}$ where $\Delta\mathbf{x}_{i+1} = \mathbf{x}_{i+1} - \mathbf{x}_i$.

2.3 Generative adversarial network for generate data

A generative model is a statistical model that can generate new data from the distribution of a real data set. Given a set of data inputs X and labels Y, a generative model would capture the joint probability $p(X, Y)$ and would be able to generate a new instance $(\mathbf{x}, \mathbf{y}) \sim p(X, Y)$. A Generative Adversarial Network (GAN) is a generative model based on deep neural-networks [8]. GAN's are

commonly used to for image processing and to generate new images for the dataset. Nevertheless, we propose the use of GAN to generate one-dimensional samples that describe the motion of an agent.

GAN is comprised of two networks trained simultaneously: a *discriminator* and a *generator* networks. The generator is trained to capture the distribution of the training data and to generate plausible data of the same distribution. The discriminator, on the other hand, is trained to distinguish between real data and fake data outputted by the generator. Hence, the discriminator is able to penalize the generator for producing implausible results. In contrast, the generator attempts to maximize the probability of the discriminator to incorrectly classify either real or fake data instances. Consequently, GAN is a two-player game where both generator and discriminator try to overcome each other. Each model will have its own loss function. The generator function $G : \mathbb{R}^d \to \mathcal{C}$ maps a d-dimensional noise vector to a state vector where d is some tunable hyper-parameter. The noise vector $\mathbf{z} \in \mathbb{R}^d$ is randomly sampled from a prior normal distribution $\mathcal{N}$ and yields a generated state sample $G(\mathbf{z})$. The discriminator function $D : \mathcal{C} \to [0, 1]$ takes a state $\mathbf{x} \in \mathcal{C}$ and tries to classify whether it came from the real dataset or artificially generated. The output is single scalar denoting the certainty of the classification. The discriminator inputs come from two sources including real instances from distribution μ and fake ones created by the generator. Deriving the cross-entropy between the real and generated distributions, the loss function is defined by

$$V(D, G) = \mathbb{E}_{\mathbf{x}\sim\mu}[\log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim\mathcal{N}}[\log(1 - D(G(\mathbf{z})))] \tag{1}$$

where $\mathbb{E}$ denotes the expected value with respect to a distribution. While the training of the discriminator aims to maximize $V(D, G)$, i.e.,

$$L_D = \max_D V(D, G), \tag{2}$$

the generator requires the opposite by solving

$$L_G = \min_G V(D, G). \tag{3}$$

We note that the generator cannot directly minimize $\log(D(x))$ but only the second component in (1), i.e., $\log(1 - D(G(z)))$.

In our work, we generate states and actions of an agent that resemble real recorded data. Therefore, the generator G would generate transition data $(\mathbf{x}_t, \mathbf{a}_t, \mathbf{x}_{t+1})$ from the joint distribution of states and actions. With such data, we can train a supervised learning model $\tilde{f}$ to predict the next state, given current state and future actions, i.e., $\mathbf{x}_{i+1} = \mathbf{x}_i + \tilde{f}(\mathbf{x}_i, \mathbf{a}_i)$.

3 Results

We test our approach on two robotic systems including a micro-ground vehicle and a two-fingers underactuated hand.

Micro-Ground Robot (MGR). MGR is a two wheel drive mobile robot seen in Figure 1. The body of the MGR is fabricated by 3D printing with a Polylactic-Acid (PLA) filament. The robot size is $20 \times 40 \times 25\ mm$ and it weighs 130 grams. The MGR has an Aruco marker with a unique ID such that a camera can identify the vehicle and acquire its state $\mathbf{x}_t \in SE(2)$ relative to some world coordinate frame in real-time. An action $\mathbf{a}_t \in \mathbb{R}^2$ of the robot is the required angle changes for the two wheels during some constant time step. Furthermore, we assume that the surface in which the agent moves on is uniform such that its transition does not depend on the current state. Therefore, the transition model depends solely on the action $\Delta x_t = f(\mathbf{a}_t)$ where $\Delta x_t = (\Delta x_t, \Delta y_t, \Delta\theta_t)^T$ is the state change relative to the MGR coordinate frame at time t. We have used two MGR's while one is used for data collection and the other is used only for generating test trajectories. Collected data is comprised of $N = 55,068$ labeled data points from various trajectories.

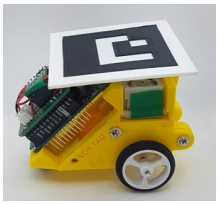


Figure 1: Micro-Ground Robot (MVP).

Underactuated Hand (UH). We consider a two-fingered adaptive hand [12] comprised of two opposing tendon-based fingers as seen in Figure 2. Each finger has two compliant joints with springs where a tendon wire runs along its length and connected to an actuator. Also, each distal link of the finger has high friction pads to avoid slipping. The observable state $\mathbf{x}$ of the hand is composed of the

Table 1: Accuracy comparison of different models for MGR trained on real and synthetic data

Model	Real data		Synthetic data	
	Pos. error (mm)	Orientation error (°)	Pos. error (mm)	Orientation error (°)
FC-ANN	0.167	0.098	0.179	0.094
CNN	0.248	0.103	0.224	0.099
LSTM	0.101	0.088	0.105	0.086
GRU	0.102	0.083	0.098	0.067

Table 2: Accuracy comparison of different models for UH trained on real and synthetic data

Model	Real data	Synthetic data
	Position error (mm)	
FC-ANN	0.112	0.115
CNN	0.291	0.317
LSTM	0.103	0.101
GRU	0.109	0.107

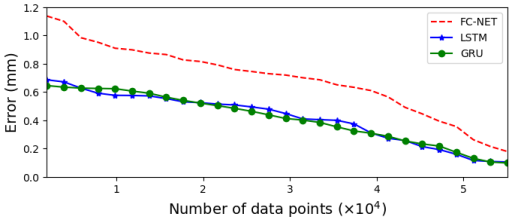


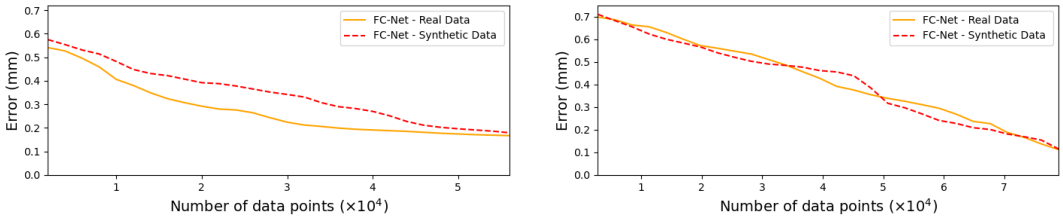
Table 3: Transition model accuracy trained on synthetic MGR data with regards to the size of data used to train the GAN.

position of the object (measured with a motion capture system) and actuator loads. Action $\mathbf{a}$ is the change of actuator angles (i.e., pulling or releasing the tendons) over a fixed time step Δt. We collect data during the manipulation of a 30 mm cylinder. A similar transition model for this system was initially introduced in [18].

For each system, a GAN was trained with the recorded data. We present a comparison of common supervised learning models including Fully-connected ANN (FC-ANN), Convolutional Neural-Network (CNN), Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) [4]. All models were optimized to yield the lowest loss value and evaluated on the test data. Tables 1 and 2 show MGR and UH results, respectively, for average one-step prediction error while training the models using real data (without using GAN) and synthetic data. Figure 3 shows the accuracy of the models with regards to the size of real data used to train the GAN. We also tested the LSTM models for the UH with test data of an ellipse cross-section prism. The accuracy was 0.111 mm and 0.102 mm for models of real and synthetic data, respectively.

Figure 2: A two-finger underactuated hand manipulating a cylinder.

Results show that synthetic data is at least equivalent to real data and can even improve accuracy. Figures 3a and 3b validate this be showing the accuracy with regards to data size, for MGR and UH, respectively. Furthermore, the generated data can also deal with some changes in the system such as a different MGR or object replacement of the UH. In conclusion, the results show that training a generative model can provide synthetic data that is sufficient for an accurate transition model with robustness to uncertainties. By having the generative model available open-source along with the hardware, any user can build multiple robots and easily generate synthetic data for modeling and simulating them.



(a) (b)

Figure 3: Position accuracy of the FC-ANN trained on real and synthetic data for (a) MGR and (b) UH, with regards to the data size.

References

[1] Osher Azulay and Amir Shapiro. Wheel loader scooping controller using deep reinforcement learning. *IEEE Access*, 9:24145–24154, 2021.

[2] Sukarna Barua, Sarah Monazam Erfani, and James Bailey. Fcc-gan: A fully connected and convolutional net architecture for gans. *arXiv preprint arXiv:1905.02417*, 2019.

[3] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks. *arXiv preprint arXiv:1810.10863*, 2018.

[4] Rahul Dey and Fathi M. Salem. Gate-variants of gated recurrent unit (gru) neural networks. In *IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1597–1600, 2017.

[5] Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.

[6] Tomer Golany, Daniel Freedman, and Kira Radinsky. Ecg ode-gan: Learning ordinary differential equations of ecg dynamics via generative adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 134–141, 2021.

[7] Tomer Golany, Kira Radinsky, and Daniel Freedman. Simgans: Simulator-based generative adversarial networks for ecg synthesis to improve deep ecg classification. In *International Conference on Machine Learning*, pages 3597–3606. PMLR, 2020.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the International Conference on Neural Information Processing Systems*, volume 2, page 2672–2680. MIT Press, 2014.

[10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[11] Teguh Santoso Lembono, Emmanuel Pignat, Julius Jankowski, and Sylvain Calinon. Learning constrained distributions of robot configurations with generative adversarial network. *IEEE Robotics and Automation Letters*, 6(2):4233–4240, 2021.

[12] Raymond R. Ma and Aaron M. Dollar. Yale openhand project: Optimizing open-source hand designs for ease of fabrication and adoption. *IEEE Rob. & Aut. Mag.*, 24:32–40, 2017.

[13] Keita Matsumoto, Atsushi Yamaguchi, Takahiro Oka, Masahiro Yasumoto, Satoru Hara, Michitaka Iida, and Marek Teichmann. Simulation-based reinforcement learning approach towards construction machine automation. In *Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC)*, pages 457–464, 2020.

[14] Blazej Osinski, Adam Jakubowski, Piotr Milos, Pawel Ziecina, Christopher Galias, S. Homoceanu, and Henryk Michalewski. Simulation-based reinforcement learning for real-world autonomous driving. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6411–6418, 2020.

[15] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2018.

[16] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[17] Lars Ruthotto and Eldad Haber. An introduction to deep generative modeling. *GAMM-Mitteilungen*, 44(2):e202100008, 2021.

[18] A. Sintov, A. S. Morgan, A. Kimmel, A. M. Dollar, K. E. Bekris, and A. Boularias. Learning a state transition model of an underactuated adaptive hand. *IEEE Robotics and Automation Letters*, 4(2):1287–1294, 2019.

[19] Jingjin Yu, Shuai D. Han, Wei N. Tang, and Daniela Rus. A portable, 3d-printing enabled multi-vehicle platform for robotics research and education. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1475–1480, 2017.

[20] Wenshuai Zhao, J. P. Queralta, and T. Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. *IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744, 2020.

[21] Zhengli Zhao, Zizhao Zhang, Ting Chen, Sameer Singh, and Han Zhang. Image augmentations for gan training. *arXiv preprint arXiv:2006.02595*, 2020.

[22] Fei Zhu, Fei Ye, Yuchen Fu, Quan Liu, and Bairong Shen. Electrocardiogram generation with a bidirectional lstm-cnn generative adversarial network. *Scientific reports*, 9(1):1–11, 2019.

[23] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.