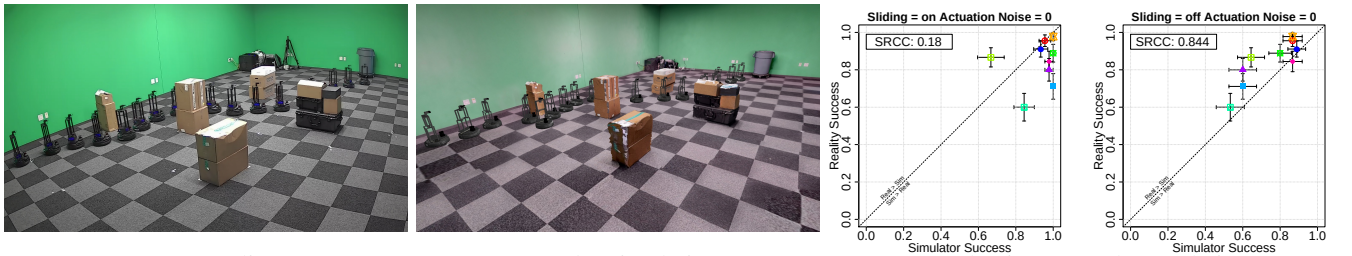
Sim2Real Predictivity: Does Evaluation in Simulation Predict Real-World Performance?

Abhishek Kadian[1]*, Joanne Truong[2]*, Aaron Gokaslan[1], Alexander Clegg[1], Erik Wijmans[1,2]
Stefan Lee[3], Manolis Savva[1,4], Sonia Chernova[1,2] Dhruv Batra[1,2]
[1]Facebook AI Research [2]Georgia Institute of Technology [3]Oregon State University [4]Simon Fraser University



(a) Reality. (b) Simulation. (c,d) Sim-vs-Real Correlation (SRCC).

Fig. 1: We measure the correlation between visual navigation performance in simulation and in reality by virtualizing reality and executing parallel experiments. (a): Navigation trajectory in a real space with obstacles. (b): Virtualized replica in simulation. (c): $\text{SRCC}_{\text{Succ}}$ for AI Habitat Challenge 2019 setting, showing a relatively low correlation between real and simulated performance. (d) By optimizing for $\text{SRCC}_{\text{Succ}}$, we arrive at simulation settings that are highly predictive of real-world performance.

***Abstract*— Does progress in simulation translate to progress on robots? If one method outperforms another in simulation, how likely is that trend to hold in reality on a robot? We examine this question for embodied PointGoal navigation – developing engineering tools and a research paradigm for evaluating a simulator by its *sim2real predictivity*.**

First, we develop Habitat-PyRobot Bridge (HaPy), a library for seamless execution of identical code on simulated agents and robots – transferring simulation-trained agents to a LoCoBot platform with a one-line code change. Second, we investigate the sim2real predictivity of Habitat-Sim [1] for PointGoal navigation. We 3D-scan a physical lab space to create a *virtualized replica*, and run parallel tests of 9 different models in reality and simulation. We present a new metric called Sim-vs-Real Correlation Coefficient (SRCC) to quantify predictivity.

We find that SRCC for Habitat as used for the CVPR19 challenge is low (0.18 for the success metric), suggesting that performance differences in this simulator-based challenge do not persist after physical deployment. This gap is largely due to AI agents learning to exploit simulator imperfections – abusing collision dynamics to 'slide' along walls , leading to shortcuts through otherwise non-navigable space. Naturally, such exploits do not work in the real world. Our experiments show that it is possible to *tune* simulation parameters to improve sim2real predictivity (*e.g.* improving $\text{SRCC}_{\text{Succ}}$ from 0.18 to 0.844) – increasing confidence that in-simulation comparisons will translate to deployed systems in reality.

I. Main

All simulations are wrong, but some are useful.

A variant of a popular quote by George Box

The vision, language, and learning communities have recently witnessed a resurgence of interest in studying integrative robot-inspired agents that perceive, navigate, and interact with their environment. Such work has commonly been carried out in simulation rather than in real-world environments. Simulators can run orders of magnitude faster than real-time [1], can be highly parallelized, and enable *decades* of agent experience to be collected in days [2]. Moreover, evaluating agents in simulation is safer, cheaper, and enables easier benchmarking of scientific progress than running robots in the real-world.

However, no simulation is a perfect replica of reality, and AI systems are known to exploit imperfections and biases to achieve strong performance in simulation which may be unrepresentative of performance in reality. Notable examples include evolving tall creatures for locomotion that fall and somersault instead of learning active locomotion strategies [3] and OpenAI's hide-and-seek agents abusing their physics engine to 'surf' on top of obstacles [4].

This raises several fundamental questions of deep interest to the scientific and engineering communities: *Do comparisons drawn from simulation translate to reality for robotic systems?* Concretely, if one method outperforms another in simulation, will it continue to do so when deployed on a robot? Should we trust the outcomes of embodied AI challenges (*e.g.* the AI Habitat Challenge at CVPR 2019) that are performed entirely in simulation? These are questions not only of simulator *fidelity*, but rather of *predictivity*.

In this work, we examine the above questions in the context

of PointGoal Navigation (PointNav) [5] with Habitat and the LoCoBot robot [6] as our simulation and reality platforms – focusing on measuring and optimizing the predictivity of a simulator. High predictivity enables researchers to use simulation for evaluation with confidence that the performance of different models will generalize to real robots. We introduce engineering tools and a research paradigm for performing simulation-to-reality (sim2real) indoor navigation studies, revealing surprising findings about prior work.

Habitat-PyRobot Bridge: Simple Sim2Real. We introduce the Habitat-PyRobot Bridge (HaPy), a software library that enables seamless sim2robot transfer. As the name suggests, HaPy integrates Habitat [1], a high-performance, photorealistic 3D simulator, with PyRobot [7], a recently released high-level API that implements simple interfaces to abstract lower-level control and perception for mobile platforms (LoCoBot), and manipulators (Sawyer), and offers a seamless interface for adding new robots. HaPy makes it trivial to *execute identical code in simulation and reality*, and is able to benefit from the scalability and generalizability of both Habitat and PyRobot. We will open-source HaPy so that everyone has this ability.

Evaluation Environment. We prepare a real lab space, called CODA, within which the robot must navigate while avoiding obstacles. We *virtualize* this lab space (under different obstacle configurations) using a Matterport Pro2 3D camera to collect 360° scans at multiple points in the room, ensuring full coverage. These scans are used to reconstruct 3D meshes of the environment which can be directly imported into Habitat. This streamlined process is easily scalable and enables quick virtualization of new physical spaces.

Sim2Real Correlation Coefficient. Our notable thesis is that simulators need not be a perfect replica of reality to be useful. Specifically, we should primarily judge simulators not by their visual or physical realism, but by their *sim2real predictivity* – if method A outperforms B in simulation, how likely is the trend to hold in reality? We propose the use of a quantity we call Sim-vs-Real Correlation Coefficient (SRCC) to quantify the degree to which performance in simulation translates to performance in reality.

Let (s_i, r_i) denote accuracy (episode success rate, SPL [8], *etc.*) of navigation method i in simulation and reality respectively. Given a paired dataset of accuracies for n navigation methods $\{(s_1, r_1), \ldots, (s_n, r_n)\}$, SRCC is the sample Pearson correlation coefficient (bivariate correlation).[1]

SRCC values close to $+1$ indicate high linear correlation and are desirable, insofar as changes in simulation performance metrics correlate highly with changes in reality performance metrics. Values close to 0 indicate low correlation and are undesirable as they indicate changes of performance in simulation are not predictive of real world changes in performance. Beyond the utility of SRCC as a simulation predictivity metric, we can also view it as an optimization objective for simulation parameters.

Concretely, let θ denote parameters controlling the simulator (amount of actuation noise, lighting, *etc.*). We can view simulator design as optimization problem: $\max_\theta \text{SRCC}(S_n(\theta), R_n)$ where $S_n(\theta) = \{s_1(\theta), \ldots, s_n(\theta)\}$ is the set of accuracies in simulation with parameters θ and R_n is the same performance metric computed on equivalent episodes in reality. Note that θ affects performance in simulation $S_n(\theta)$ but not R_n since we are only changing test-time parameters. The specific navigation models themselves are held fixed. Overall, this gives us a formal approach to simulator design instead of operating on intuitions and qualitative assessments.

In contrast, if a simulator has low SRCC but high mean real world performance, researchers will not be able to use this simulator to make decisions (e.g. model selection) because they can't know if changes to performance in simulation will have a positive or negative effect on real-world performance. Every change will have to be tested on the physical robot.

Measuring the Sim2Real Gap. We train and evaluate 9 different learning-based navigation models for PointGoal in Habitat, using the Gibson dataset [9] for training. Agents are trained from scratch with reinforcement learning using DD-PPO [2] – a decentralized, distributed proximal policy optimization [10] algorithm that is well-suited for GPU-intensive simulator-based training.

We find that SRCC for Habitat as used for the CVPR19 challenge is 0.603 for the Success weighted by (normalized inverse) Path Length (SPL) metric and 0.18 for agent success. When ranked by SPL, we observe 9 relative ordering reversals from simulation to reality, suggesting that the results/winners may not be the same if the challenge were run on LoCoBot.

We find that large-scale RL trained models can learn to 'cheat' by exploiting the way Habitat allows for 'sliding' along walls on collision. Essentially, the virtual robot is capable of cutting corners by sliding around obstacles, leading to unrealistic shortcuts through parts of non-navigable space and 'better than optimal' paths. Naturally, such exploits do not work in the real world where the robot stops on contact with walls.

We *optimize* SRCC over Habitat design parameters and find that a few simple changes improve SRCC_{SPL} from 0.603 to 0.875 and $\text{SRCC}_{\text{Succ}}$ from 0.18 to 0.844. The number of rank reversals nearly halves to 5 (13.8%). Furthermore, we identify highly-performant agents in *both* this new simulation and on LoCoBot in real environments.

While our experiments are conducted on PointNav, we believe our software (HaPy), experimental paradigm (sim2real predictivity and SRCC), and take-away messages will be useful to the broader community.

Please refer to our arxiv paper and project page for more information.

[1] Other metrics such as rank correlation can also be used.

REFERENCES

[1] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, *et al.*, "Habitat: A Platform for Embodied AI Research," in *ICCV*, 2019. 1, 2

[2] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, *et al.*, "DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames," in *ICLR*, 2020. 1, 2

[3] J. Lehman, J. Clune, D. Misevic, C. Adami, L. Altenberg, *et al.*, "The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities," *arXiv:1803.03453*, 2018. 1

[4] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, *et al.*, "Emergent tool use from multi-agent autocurricula," *arXiv preprint arXiv:1909.07528*, 2019. 1

[5] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, *et al.*, "On Evaluation of Embodied Navigation Agents," *arXiv preprint arXiv:1807.06757*, 2018. 2

[6] "Locobot: An open source low cost robot," https://locobot-website.netlify.com/, 2019. 2

[7] A. Murali, T. Chen, K. V. Alwala, D. Gandhi, L. Pinto, *et al.*, "Pyrobot: An open-source robotics framework for research and benchmarking," *arXiv preprint arXiv:1906.08236*, 2019. 2

[8] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, *et al.*, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *CVPR*, 2018. 2

[9] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson env: Real-world perception for embodied agents," in *CVPR*, 2018. 2

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. 2

*Denotes equal contribution.