1 Preliminary Work-Packages

2 Test-data and model generation

This section covers our ideas on how to generate different models and test-data. To tackle this task we asked ourselves the following questions:

1. How to generate test-data?
2. How to generate distribution shifts? How much shift (how to measure it)?
3. How to train distinct models that perform equally well on test data?
4. How do you know that two models are distinct?
5. How do you know that they perform equally well on the test data?

In the following our ideas a clustered to answer these questions and enhanced by ideas from literature / other researchers had when tackling similar tasks.

2.1 Test-data generation

Idea: Gain real experience sampled from the real environment. *trueMDP*

- enables us to mitigate possible training errors w.r.t. shortcuts an agent can take by exploiting errors in or simplicity of the simulation itself
- enables us to prioritize scenarios actually happen in the real environment
- we'll have to mitigate the overhead for the real environment created by data sampling process
- we'll have to make sure not to create site-effects within the real environment

Idea: Define a simulation that models the environment so we can generate test data without the drawbacks of sampling from the environment. *approximateMDP*

- enables us to to generate test data fast
- enables us to create observations independently, no matter whether they occurred often, rarely, not yet in the real environment
- no consequences in the real environment

For mRUBiS: We might go with a hybrid approach. First, setup scenarios and inject issues to the real environment and observe the states and actions taken. Secondly, learn the transitions stochastically and build a model of it, that can later be used to generate new observations.
Related ideas from the literature: When it comes to transitioning between simulation and real environments there are a couple of tools and techniques already explored by researchers working in the field of robotics and autonomous vehicle driving. We may can leverage some of the following ideas:

Distributed rollouts „Algorithms that use distributed rollouts, where multiple simulations are executed in parallel to collect experience data, were introduced to reduce training time“[1, p. 2] (They reference the respective papers => we can look into them in case we want to use this technique.)

domain randomization „For a successful transfer to the real world, researchers use calibration [2], [22], domain randomization [23], [24], [25], [12], fine tuning with real world data [9], and learn features from a combination of simulation and real data“[1, p. 1] During evaluation they used domain randomization (next to distributed rollouts). the goal: „We test whether robust evaluation in simulation is indicative of real world performance. If true, we can identify when to stop training in simulation and avoid underfitting/overfitting.“[1, p. 5] Kaspar et. al. also mention randomization, especially dynamics and environment randomization, even though their approach is to not rely on them. [4, p. 4] -> not 100

compact observation space „Furthermore, they find that a compact observation space is helpful for sim to real transfer, because the policy cannot overfit to unimportant details of the observation.“[4, p. 2] In the paper they leverage this idea by investigating „possibilities for sim to real transfer while trying to make the task to learn as easy as possible by using the Operational Space Control framework.“[4, p. 1] (In section four they also describe their setup within the framework but it seems to be relevant for robotic relate tasks.)

2.2 Distribution shift generation

Idea: think system dynamics as finite Markov Decision Process

- Simulate changes in the environmental behavior by changing the probabilities for each failure type. Achieve this by changing the probabilities of transitions in a Hidden Markov Model (HMM).
- To slightly generalize: change of the distribution is equivalent to a change of probabilities for transitioning from one state into another.
- For an agent a distribution shift can only be measured if it reflects in observations (HMM perspective).

2.3 Model generation

Idea: Train distinct models with equal performance on test data and monitor their performance

- sample different train datasets from same domain space
- initialize same model with different configurations
- prevent problem specialization by inducing noise

- collect train data under different times, location or state of the environment
- train models until they converge in a shared performance metric (e.g. cumulative total reward, cumulative regret)

We intend to mitigate the following challenges in model generation:

Ensure model distinction: by

- compare behavior of the models on the same input => we assume different behavior for distinct models
- compare model structure, e.g. initial state, weights and transitions

Ensure equal performance: by

- stratified performance evaluation
- contrastive evaluation

Related ideas from the literature: In software testing a technique called Model-based mutation testing can be used to evaluate up to which degree given model describes a system under consideration. [2]

Remarks by Prof. Giese: while this technique helps to identify issues in correctly modeling a desired system, different NN model configurations do not necessarily result in observable differences in the model's output. Therefore, techniques leveraging the ideas behind model-based mutation testing may not be helpful to tackle our challenges.

2.4 Other Sim to Real challenges

Related ideas from the literature: When it comes to transitioning between simulation and real environments there are a couple of tools and techniques already explored by researchers working in the field of robotics and autonomous vehicle driving. We may can leverage some of the following ideas:

System Identification Mitigate the problem when the dynamics in the real environment are too different from the ones in simulation and therefore transferred policy performs poorly in real environment. „The dynamics of the real robot were too different from the dynamics of the simulation. Therefore, we performed a special type of system identification, where we run scripted trajectories of actions a_t for n timesteps on the real robot. Then we used the [...] algorithm to change the simulation parameters."[4, p. 4]

notion of close-loop nature „Most of the previous work to test and verify systems with ML components focuses only on the ML components themselves, without consideration of the closed-loop behavior of the system. [...] The iterative process of sensing, processing, and actuating is what we refer to as closed-loop behavior."[5, p. 2] Their main concern is for a automobile to interact with the environment and e.g. identify situations were an accident can be prevented (within car's control) or not (within control of other cars or pedestrians or ...) -> we do not have the same situation for our shop system but maybe situations were we have a performance problem because some of our components went down vs. we are under a (D)DoS attack or something like that? (In case that's even within our project scope....)

Sim2Real Correlation Coefficient (SRCC) is a metric to quantify predictability. „Let (s_i, r_i) denote accuracy of navigation method i 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."[3, pp. 5]

References

[1] B. Balaji, S. Mallya, S. Genc, S. Gupta, L. Dirac, V. Khare, G. Roy, T. Sun, Y. Tao, B. Townsend, E. Calleja, S. Muralidhara, and D. Karuppasamy. "DeepRacer: Educational Autonomous Racing Platform for Experimentation with Sim2Real Reinforcement Learning". In: *CoRR* abs/1911.01562 (2019). arXiv: `1911.01562`.

[2] F. Belli, C. J. Budnik, A. Hollmann, T. Tuglular, and W. E. Wong. "Model-based mutation testing—Approach and case studies". In: *Science of Computer Programming* 120 (2016), pages 25–48. ISSN: 0167-6423. DOI: `https://doi.org/10.1016/j.scico.2016.01.003`.

[3] A. Kadian, J. Truong, A. Gokaslan, A. Clegg, E. Wijmans, S. Lee, M. Savva, S. Chernova, and D. Batra. "Are We Making Real Progress in Simulated Environments? Measuring the Sim2Real Gap in Embodied Visual Navigation". In: *CoRR* abs/1912.06321 (2019). arXiv: `1912.06321`.

[4] M. Kaspar, J. D. M. Osorio, and J. Bock. "Sim2Real Transfer for Reinforcement Learning without Dynamics Randomization". In: *CoRR* abs/2002.11635 (2020). arXiv: `2002.11635`.

[5] C. E. Tuncali, G. Fainekos, H. Ito, and J. Kapinski. "Simulation-based Adversarial Test Generation for Autonomous Vehicles with Machine Learning Components". In: *CoRR* abs/1804.06760 (2018). arXiv: `1804.06760`.