IN2349 ADLR: Project Ideas

October 19, 2023

Here you can find several ideas for projects we collected. Take this as an inspiration for your project. Some ideas are rather "big", meaning they could result in multiple projects. After registering your team (including a draft proposal), you will discuss the extent of your final proposal with your assigned tutor.

1 Closing the Sim-to-Real Loop: Adapting Simulation Randomization with Real World Experience

In Chebotar et al. [4] an efficient method for solving the sim2real problem by iteratively adapting the simulation parameters to the real system is proposed.

- Create an experimental sim2sim setup because no real robot is available (i.e., try to adapt one simulation to a given one with unknown parameters).
- Re-implement their algorithm based on the relative entropy policy search or using reinforcement learning algorithms.
- Investigate the convergence of estimated parameters.
- Investigate the influence of a broader starting distribution on the final performance.
- Investigate how re-using learned policies from previous iterations for initialization of the new network shortens training time and restrains the final performance.

2 Non-sequential Reinforcement Learning for Hard Exploration Problems

A way of dealing with sparse reward signals is to utilize "expert demonstrations" in promising regions of the state space. In the absence of experts, Blau et al. [3] propose to first generate successful trajectories by RRT-based planning algorithms and then initialize a policy using that data. Based on this idea, you could experiment with:

- Implementing different heuristics for choosing states to explore from (see e.g. Ecoffet et al. [6]) or implementing an entirely different planning algorithm.
- Coupling the policy learning with the RRT-based data collection e.g. by sampling starting states according to RRT and executing actions according to the current policy

3 Tactile exploration of objects

For grasping an object with a robotic hand, often a full 3D model of the given object is required. Using (depth-)camera, one is able to infer the surface of the visible part of the object, however also the surface of the occluded parts is usually needed to, e.g., robustly grasp the object. With tactile exploration (i.e., slowly moving the robotic hand until the fingers touch the object), it is possible to observe even the occluded parts of the object. This tactile information is usually very sparse. The idea is to use a learning-based approach to complete the full shape from this sparse contact information.

- Start in 2D: Train a neural network to complete a 2D shape based on a few given points similar to Watkins-Valls et al. [27] did in the more complex 3D case.
- Develop a strategy which determines the best way for the next tactile exploration based on previous points using reinforcement learning.
- Bring it into 3D: Complete 3D shapes in the same way as done for 2D
- Combine tactile and visual depth information to infer the objects shape

4 Generative networks for robot grasping

Generative neural networks can be used to directly generate stable grasps for a given unkown object. For a parallel jaw gripper, the network learns a distribution over 6D end effector poses conditioned on the observed object. For this application, different architectures have been evaluated, including Variational Autoencoder, Generative Adversarial Networks and autoregressive architectures Winkelbauer et al. [29]. Recently, diffusion-based architectures [12] have shown exciting results in image generation. The goal of this project is to apply diffusion-based architectures to the problem of grasp generation.

- For training, we make use of the existing public training dataset Acronym [7].
- Preprocess the data and make it ready for our learning approach.
- Adapt the diffusion architecture for robotic grasping.
- Compare the results with an existing architecture.
- Optional: Extend the approach to generating grasps constraint to lie on a specified part of the object [21].

5 Tactile Material Classification

Tulbure and Bäuml [26] (as mentioned in the lecture) proved that one can classify a vast number of materials robustly ($\approx 95\%$) using only the spatio-temporal signal of a tactile skin.

- You will get the data of Tulbure and Bäuml [26]
- Often it is way easier to collect unlabelled samples than labelled. Can we learn a representation without the labels? Recent work in this direction is, e.g. BYOL (Grill et al. [8]) or a Joint Embedded Predictive Architecture (Assran et al. [2]).

- Maybe you can improve the classification performance and "beat" Tulbure and Bäuml [26] by using hyperparameter and neural architecture search. You find the methods in the lecture 3 of the course.

6 Unsupervised Skill Discovery / Curiosity

Pretraining neural networks in an unsupervised setting showed to be extremely effective for language models. Models such as GPT or Bert can be fine tuned or used directly on downstream tasks. Similarly, RL agents can be pretrained in an environment without an extrinsic reward signal and later be adapted to specific tasks. Laskin et al. [17] compare many different approaches on a unified benchmark (URLB).

- Skim recent literature on unsupervised RL (e.g. [10, 18, 20])
- Choose one/come up with your own ideas or modifications.
- Implement and compare them with the results reported by URLB.

7 Learning the Inverse Kinematics

Look at the possibilities for representing inverse problems with neural networks. Ardizzone et al. [1] compare different flavors of GANs, VAEs and INN(theirs) for inverse problems in general. Extend their simple robotic example of a planner arm to 3D, more DoFs, or multiple TCPs. Unlike in computer vision, for the robot kinematic we have solid metrics to describe how well the generation task was performed. How can we use this knowledge to our advantage?

- What is the best approach to represent the high dimensional nullspaces for complex robot geometries?
- Lembono et al. [19] use an ensemble of GANs to reduce the mode collapse. What other options do we have to improve the generative model?
- How to measure the performance if the real nullspace is not known?
- Predict not only the position of the TCP but also its rotation. How can one best represent the $SO(3)$?

8 Motion Planning with Diffusion

Look into generative models for robotic motion planning based on the ideas work by Janner et al. [13]. The core of their approach lies in a diffusion probabilistic model that plans by iteratively denoising trajectories.

- Use a simple 2D robot to get familiar with the diffusion approach in the robotic context.
- Include the changing environment as central part of the planning problem
- Extend the framework to more complex robots

9 Trajectory Planning with Moving Obstacles

Drones not only have to plan flight paths through static environments, but also avoid collisions with dynamic objects. To learn such trajectories, a suitable encoding of the changing environment is crucial. Start with the Basis Points Set Prokudin et al. [22] and extend it to dynamic environments. Use this representation for neural motion planning Qureshi et al. [23].

- Come up with a state representation for dynamic environments.
- Set up a simple 2D (and later 3D) environment in which an agent can navigate through moving obstacles.
- Use RL to plan optimal trajectories in this environment.
- Optional: Extend the method to work with uncertainties in the motion prediction of the collision objects.

10 Learning to Fly Beyond RL – Analytic Policy Gradient

Video introduction at: https://flyonic.de/in2349/.
Fixed-wing VTOL (Vertical Take-Off and Landing) drones combine the ability to take off vertically and exploit lift for efficient cross-country flight. The disadvantage is more complicated control. Learned controllers are a promising solution here. Wiedemann et al. [28] presents a method to directly exploit the differentiability of models and thus skip the detour via reinforcement learning.

- Setting up the learning pipeline with a simple quadrotor model (simple control).
- Transfer the approaches to a VTOL model (complicated control)
- Optional: Replace the policy with a PID controller. Exploit the pipeline for automatic parameter tuning.

A Julia framework is provided for this project, including the differentiable models. Julia [14] is a modern programming language that has the speed of C but is as easy to program as Python.

11 Recurrent Off-Policy Reinforcement Learning in POMDPs

In partially observable Markov decision processes (POMDPs), an RL agent has to be equipped with some sort of memory in order to be able to act optimally. A well known method addressing this issue is to encode the history of observations by recurrent neural networks (RNNs). For the class of off-policy methods, Heess et al. [11] combine RNNs with the DDPG algorithm and Kapturowski et al. [15] study the interplay of DQN-based algorithms with recurrent experience replay. Based on this work:

- Choose or implement environments that require the use of memory to be solved optimally.
- Implement a recurrent version of the SAC algorithm by Haarnoja et al. [9].

- Assess the effect of different design choices and hyperparameters (e.g. hidden state initialization strategy in the experience replay buffer, truncated BPTT, ...)

12 Differentiable Bayesian Filters

Prior knowledge of Bayesian filters highly recommended
Differentiable filters are a promising approach for combining the algorithmic structure of bayesian filter techniques with the power of learning-based methods (for an overview over existing methods, see e.g. Kloss et al. [16]). Importantly, differentiable filters offer a systematic way of dealing with aleatoric uncertainty in state estimation problems.

- Implement a *differentiable* filter of your choice, for example EKF, UKF or Particle Filter.
- Consider the simple tracking experiment VI from Kloss et al. [16]. Implement interesting modification to the experiment. For example:
  - use the distance to (a subset of) beacons instead of images as measurements
  - implement collisions
  - additionally, estimate parameters from the system dynamics on-the-fly
- Compare your filter to recurrent neural networks and discuss the pros and cons.

13 Diffusion Policies

Diffusion-based architectures have shown exciting results in image generation. Recently, researchers started to apply them to policy learning. Chi et al. [5] show promising imitation learning results with both vision- and state-based policies.

- Apply the diffusion policy [5] to simple benchmark environments with expert offline data (link).
- Analyze if the approach is better suited for specific types of environments (e.g. planning vs. control).
- Adapt the diffusion architecture for other types of environments.
- Optional: Develop and implement ideas on how the architecture can be used in a reinforcement learning setting.

14 Casting Sim2Real as Meta-Reinforcement Learning

The PEARL algorithm introduced by Rakelly et al. [24] and the RL^2 algorithm evaluated by Yu et al. [30] promise sample-efficient meta-reinforcement learning, meaning that the algorithm can quickly adapt to new unseen tasks. We would like use the capabilities to handle heavily randomized environments occurring in simulations that are designed to allow a real-world transfer of the policy.

- Find a suitable benchmark environment and select an algorithm (together with the tutor).
- Implement the algorithm on top of an existing basic RL algorithm (e.g. SAC or PPO) and compare the performance.
- Examine the reward for constant as well as dynamically changing environments.
- Analyze different loss terms and evaluate the performance.
- Extend the algorithm to deal with problems occurring in Sim2Sim settings with unknown disturbances.

15 Information Bottleneck / Ignoring Noise

The reinforcement learning framework allows us to specify arbitrary observation spaces. For robotic tasks in particular, often we have an intuitive understanding of what observations might be necessary to solve a given task. However, it is usually unclear what additional (easily available) information is beneficial for training times and real world performance. For practical purposes it would be convienent to pass all the available information to the agent. This raises serveral questions:

- Can we pass too much information?
- Is redundant information harmful or maybe beneficial (similar to over-parameterization)?
- Can the agant learn to ignore noise inputs?

Explore those questions on simple environments (link) and come up with network architectures, such as self-attention (cf. Tang et al. [25]) or autoencoders, to fix arising problems.

References

[1] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W Pellegrini, Ralf S Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*, 2018.

[2] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. 2023.

[3] Tom Blau, Philippe Morere, and Gilad Francis. Learning from demonstration without demonstrations. In *IEEE International Conference on Robotics and Automation*, 2021.

[4] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan D. Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. *CoRR*, 2018.

[5] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[6] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

[7] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. Acronym: A large-scale grasp dataset based on simulation. In *IEEE International Conference on Robotics and Automation*, 2021.

[8] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent a new approach to self-supervised learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

[9] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, 2018.

[10] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains through world models. *CoRR*, abs/2301.04104, 2023.

[11] Nicolas Heess, Jonathan J Hunt, Timothy P Lillicrap, and David Silver. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Neural Information Processing Systems*, 2020.

[13] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[14] Julia. Julia. `http://julialang.org`, 2022.

[15] Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations*, 2018.

[16] Alina Kloss, Georg Martius, and Jeannette Bohg. How to train your differentiable filter. *Autonomous Robots*, pages 1–18, 2021.

[17] Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. URLB: unsupervised reinforcement learning benchmark. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1*, 2021.

[18] Michael Laskin, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel. Unsupervised reinforcement learning with contrastive intrinsic control. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 34478–34491. Curran Associates, Inc., 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/debf482a7dbdc401f9052dbe15702837-Paper-Conference.pdf`.

[19] 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.

[20] Mengdi Li, Xufeng Zhao, Jae Hee Lee, Cornelius Weber, and Stefan Wermter. Internally rewarded reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 20556–20574. PMLR, 2023.

[21] Jens Lundell, Francesco Verdoja, Tran Nguyen Le, Arsalan Mousavian, Dieter Fox, and Ville Kyrki. Constrained generative sampling of 6-dof grasps, 2023.

[22] Sergey Prokudin, Christoph Lassner, and Javier Romero. Efficient Learning on Point Clouds with Basis Point Sets. In *International Conference on Computer Vision*, aug 2019.

[23] Ahmed Hussain Qureshi, Yinglong Miao, Anthony Simeonov, and Michael C. Yip. Motion planning networks: Bridging the gap between learning-based and classical motion planners. *CoRR*, abs/1907.06013, 2019.

[24] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[25] Yujin Tang, Duong Nguyen, and David Ha. Neuroevolution of self-interpretable agents. In Carlos Artemio Coello Coello, editor, *GECCO '20: Genetic and Evolutionary Computation Conference, Cancún Mexico, July 8-12, 2020*, pages 414–424. ACM, 2020. doi: 10.1145/3377930.3389847.

[26] Andreea Tulbure and Berthold Bäuml. Superhuman performance in tactile material classification and differentiation with a flexible pressure-sensitive skin. In *Proc. IEEE/RAS International Conference on Humanoid Robots*, 2018.

[27] David Watkins-Valls, Jacob Varley, and Peter Allen. Multi-modal geometric learning for grasping and manipulation. In *IEEE International conference on robotics and automation*, 2019.

[28] Nina Wiedemann, Valentin Wüest, Antonio Loquercio, Matthias Müller, Dario Floreano, and Davide Scaramuzza. Training efficient controllers via analytic policy gradient, 2023.

[29] Dominik Winkelbauer, Berthold Bäuml, Matthias Humt, Nils Theurey, and Rudolph Triebel. A two-stage learning architecture that generates high-quality grasps for a multi-fingered hand. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022.

[30] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *CoRR*, 2019.