IN2349 ADLR: Project Ideas

October 20, 2022

Here you can find a number of ideas for projects we collected. Take this as an inspiration for your own project. Some of the ideas are rather "big", meaning they could result in more than one project. After you have registered 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. [2] an efficient method for solving the sim2real problem by iteratively adapting the simulation parameters to the real system is proposed. Since the physics simulations like pybullet, MuJuCo or Isaac Gym provide no gradient they need to use relative entropy policy search Peters et al. [13].

- 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 you know.
- 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 Solving Complex Sparse Reinforcement Learning Tasks

When defining Reinforcement Learning Tasks for robots, it is often desirable to stick to sparse rewards in order to avoid reward shaping. Not only does is it ease the setup since a task can be solely defined by its final desired outcome, but the optimal policy can be found "outside of the box" without a human prior given through the reward function. Unfortunately, in big state spaces random exploration is not able to find these sparse success signals. Therefore, Riedmiller et al. [17] introduce SAC-X.

- Implement the algorithm

- Investigate how the presented method can be used for finer and more dexterous object manipulation e.g. with a hand.
- Another option is, to apply the method to an path planning agent which often has to recover from dead-end-situations in static environments.

3 Non-sequential Reinforcement Learning for Hard Exploration Problems

Another 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. [1] 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. [4]) 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
- Modify the code and try it for different robots and environments.
- Is this expert knowledge necessary or can this also be achieved with a well designed curriculum?
- Look at modern approaches to represent the environment in which the robot moves (ie. Basis Points Set Prokudin et al. [14]; PointNet for Motion Planning Strudel et al. [21])

4 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. [14] and extend it to dynamic environments. Use this representation for neural motion planning Qureshi et al. [15].

- 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.

5 Learning to Fly

Fixed-wing VTOL (Vertical Take-Off and Landing) drones are highly efficient in long-range flight, but difficult to control during gusty landing phases. Holmes et al. [9] presented an deep learning based model-agnostic VTOL controller. With a similar goal Xu et al. [24] introduced an error convolution input enabling the learned controller to adapt for different airframes.

- Transfer one of the approaches to VTOL drones with only two propellers and control surfaces.
- Expand the learning to continuous action spaces.
- Investigate what sensor readings could be added to the state space to increase stability in gusty conditions.
- Utilize our provided drone model implemented in Julia [10] as the high-efficient, flexible and dynamic programming language of the future.

You will be provided with two Julia examples demonstrating RL in the environment, allowing a instant start even without prior Julia experience. With promising results during the midterm presentation, there is the option to attempt a sim-to-real transfer with a real UAV.

6 Recurrent Off-Policy Reinforcement Learning in POMDPs

In partially observable Markov decision processes (POMDPs), a 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. [8] combine RNNs with the DDPG algorithm and Kapturowski et al. [11] study the interplay of DQN-based algorithms with recurrent experience replay. Based on these works:

- Choose POMDP environments that require the use of memory to be solved optimally.
- Implement a recurrent version of the SAC algorithm (Haarnoja et al. [7]).
- Assess the effect of different design choices and hyperparameters (e.g. hidden state initialization strategy in the experience replay buffer, truncated BPTT, ...)

7 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. [12]). 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. [12]. 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.

8 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 are 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. [22] did in the more complex 3D case.
- Develop a strategy which determines the best way for the next tactile exploration based on previous points (this can be done 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

9 Active learning for robot grasping

To train a neural network on the mapping from a given object to a set of stable grasps, usually a big annotated training dataset is necessary. The creation of such a dataset is computationally expensive, as finding stable grasps for a training object requires trying out a large number of grasps. Also, it is not completely clear how the training objects should be selected. Many objects have similar shapes and therefore produce probably redundant training data. Active learning can be used to determine which new training samples would bring the biggest benefit in further training a network. In this way the overall amount of required training data can be reduced.

- Apply the non-generative part of the grasping network described in Winkelbauer et al. [23] to the reduced problem of predicting grasps for a parallel jaw gripper: Here, first random grasps are sampled on the surface of the object, then the network predicts a score for each of them. In the end the grasp with the highest predicted score is used.

- For training, we make use of an existing public training dataset (e.g. Eppner et al. [5])
- Reduce the training data to a small set and incrementally add training samples selected using active learning, e.g., using the method described in Gal et al. [6]
- Examine, how much the amount of training data can be reduced while still achieving a high prediction accuracy

10 Unsupervised Skill Discovery / Curiosity

Pretraining neural networks in an unsupervised setting showed to be extremely effective for language models. Similarly, RL agents can be pretrained in an environment with different goals in mind. Projects in this direction could (reimplement or) validate one of the papers below and extent their work with interesting ablation studies or algorithmic modifications.

- Sekar et al. [18]
- Sharma et al. [19]

11 Procgen Benchmark

Implement and experiment with different agents on this procedurally-generated benchmark Cobbe et al. [3]. You can compare your results with other submissions on the leaderboard (`github.com/openai/procgen`).

12 Casting Sim2Real as Meta-Reinforcement Learning

The PEARL algorithm introduced by Rakelly et al. [16] promises sample-efficient Meta-Reinforcement Learning, meaning that it can quickly adapt to new unseen tasks. We want to use its capabilities to handle heavily randomized environments occurring in simulations that are designed to allow a real-world transfer of the policy (like in our own work on in-hand manipulation Sievers et al. [20]).

- Find a suitable benchmark environment
- Implement the PEARL algorithm
- Extend it to allow for a continuous task distribution during training

References

[1] Tom Blau, Philippe Morere, and Gilad Francis. Learning from demonstration without demonstrations. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4116–4122. IEEE, 2021.

[2] 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*, abs/1810.05687, 2018.

[3] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 2048–2056. PMLR, 2020. URL `http://proceedings.mlr.press/v119/cobbe20a.html`.

[4] 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.

[5] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. ACRONYM: A large-scale grasp dataset based on simulation. In *Under Review at ICRA 2021*, 2020.

[6] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

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

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

[9] Grant Holmes, Mozammal Chowdhury, Aaron McKinnis, and Shawn Keshmiri. Deep learning model-agnostic controller for vtol class uas. pages 1520–1529, 2022. doi: 10.1109/ICUAS54217.2022.9836118.

[10] Julia. Julia. `http://julialang.org`.

[11] 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.

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

[13] Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[14] Sergey Prokudin, Christoph Lassner, and Javier Romero. Efficient Learning on Point Clouds with Basis Point Sets. In *International Conference on Computer Vision (ICCV)*, pages 4332–4341, aug 2019. URL `http://arxiv.org/abs/1908.09186`.

[15] 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. URL `http://arxiv.org/abs/1907.06013`.

[16] Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables, 2019. URL https://arxiv.org/abs/1903.08254.

[17] Martin A. Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Van de Wiele, Volodymyr Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing - solving sparse reward tasks from scratch. *CoRR*, abs/1802.10567, 2018. URL http://arxiv.org/abs/1802.10567.

[18] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *ICML*, 2020.

[19] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJgLZR4KvH.

[20] Leon Sievers, Johannes Pitz, and Berthold Bäuml. Learning purely tactile in-hand manipulation with a torque-controlled hand. In *Proc. IEEE International Conference on Robotics and Automation (ICRA), 2022, Philadelphia*, 2021. URL https://arxiv.org/abs/2204.03698.

[21] Robin Strudel, Ricardo Garcia, Justin Carpentier, Jean-Paul Laumond, Ivan Laptev, and Cordelia Schmid. Learning Obstacle Representations for Neural Motion Planning. In *Conference on Robot Learning (CoRL)*, 2020. URL http://arxiv.org/abs/2008.11174.

[22] David Watkins-Valls, Jacob Varley, and Peter Allen. Multi-modal geometric learning for grasping and manipulation. In *2019 International conference on robotics and automation (ICRA)*, pages 7339–7345. IEEE, 2019.

[23] 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 (IROS)*, 2022.

[24] Jie Xu, Tao Du, Michael Foshey, Beichen Li, Bo Zhu, Adriana Schulz, and Wojciech Matusik. Learning to fly: Computational controller design for hybrid uavs with reinforcement learning. *ACM Transactions on Graphics*, 38(42):1–12, July 2019. doi: 10.1145/3306346.3322940.