Modular RL for Real-Time Learning in Physical Environments

Per R. Leikanger
UiT - the Arctic University of Norway
Troms, Norway
Per.Leikanger@uit.no

Abstract

Reinforcement Learning is a powerful approach to a machine learning from interaction with the environment. Recent impressive achievements show the potency of combining Deep Learning with RL for board games and simple visually oriented tasks, but similar feats seem to wait for physical systems. We ask ourself for the reason(s) for this apparent hole in the science, on our path toward adaptive automata for physical domains.

Inspired by computational systems in biology, revisiting conceptual fundamentals of Reinforcement Learning, and dividing complex tasks into numerous parallel learners, we aim to lessen this effect for physical interaction. A simple demonstration of an algorithmic framework is implemented, creating an agent that learns complex reactive behavior in a continuous parameter space by tabular RL methods. We conclude with a discussion on possible implications from this work for adaptive agents in the physical world, and plausible further directions toward life-long learning.

Keywords— Reinforcement Learning, Robot, Sample Efficiency, Distributed Learning

Learning in Physical Environments

Reinforcement Learning (RL) methods rooted in Dynamic Programming (DP) have proven to be effective learning algorithms for decision making in environments with unmodelled or complex system behavior. A prime example of this would be for games, where modelling opponent behaviour with formal methods can be challenging. Multi-layered graph-based function approximation (e.g. deep networks) can be used to estimate the value of a state or a state-action pair, and the two machine learning methodologies appear to create a synergy effect toward something greater.

Physical interaction can, however, be challenging for these modern function approximation techniques due to sample-sparseness; Deep Learning or other supervised learning techniques require a large set of labelled samples to form appropriate signal approximation. The importance of safe exploration, constraints due to slow system dynamics[1], degradation with use[2], combined with requirements for large number of samples might all contribute to the limited use of DRL for physical systems (Amarjyoti, 2017).

Knowledge on how spacial orientation is represented in the hippocampus of the mammalian brain (Moser, Kropff, & Moser, 2008), might serve as inspiration when trying to create a real-time learned for physical interaction; By approaching orientation in a continuous parameter space with methods inspired by orientation in mammals, combined with the intrinsically distributed mechanism for biological computations, general value functions (GVFs) (Sutton et al., 2011), and the *Id* concept from Psychoanalysis (Freud, 1940), we aim to creating a distributed agent inspired by this beautiful complex system comprised of very simple building blocks. We argue further that the resulting orientation can be seen as being general for any N-dimensional continuous parameter space, and show a simple example of how learning can happen individually across state spaces.

Theory

Tabular Q-learning(Watkins, 1989) is a good entry point for RL due to its comprehensible approach to learning(adapt decisions to experience). Here, a scalar representing the quality of doing an action in a given state is saved for later to be used for action selection. The Q-value is updated according to the *Bellman equation*

$$Q(s_n, a) \leftarrow (1-\alpha)Q(s_n, a) + \alpha[r_n + \gamma \max_a Q(s_{n+1}, a)]$$

For every passing through any state s_n, the quality of an action a is updated with the immediate reward (r_n), potential future reward $\max_a Q(s_{n+1}, a)$,

For large state spaces or complicated reward schemes, this algorithm can be difficult to apply directly due to the number of visits necessary for a good policy representation. The number of samples(visits) necessary is shown to be exponential with the number of states(Sutton & Barto, 1998), and function approximation by modern multi-layered graph-based function approximation techniques can be utilized to remedy this. Deep learning have on multiple occasions proved capable for generating agents with good policies, particularly in games or simple digital environments with huge state sets.

For physical systems, it can, however, be difficult to gather the number of labelled examples necessary for proper value function approximation as seen in prominent works in the RL literature. Considering one of the largest feats in Deep Learning RL, AlphaGo, where we read about millions of complete roll-outs to train the policy network of the agent (Silver et al., 2016), similar

[1] All physical actuation can be considered slow relative to what is possible for purely digital environments, primarily being limited by available computational power.

[2] due to wear-and-tear

feats by this technology seems improbable for physical systems. Picking up an objects with uneven edges seem to be the state of the art for learned physical interaction under these technologies; Complex policies require more samples, a likely reason why deep RL approaches fail to handle elaborate policies for robotics (Amarjyoti, 2017);

Toward Physical Learning Efficiency

On the path toward learning for physical interraction, learning efficiency is perhaps the most important barrier for applicable learning systems. Aiming to remedy this, consider a simple, effective method for increasing sample efficiency by letting multiple training challenges run in parallel. We could, for example, take inspiration from the Horde architecture from (Sutton et al., 2011); By having multiple *sub*-agents ("daemons"), each with its own goal and corresponding value function, off-policy learning can happen simultaneously and individually for all. If we let the agent's policy be formed as an aggregate over all daemons' policy, where each daemon considers a much simpler challenge than for an equivalent monolithic agent, this could further decrease learning time if sub-policies can be combined to create a coherent super-policy.

Separation of Concerns (van Seijen, Fatemi, Romoff, & Laroche, 2016), and Hybrid Reward Architecture (HRA) (Van Seijen et al., 2017) take up this notion and have looked at similar concepts for splitting up the state space into smaller sub-problems for the game of Ms. PacMan. One could take inspiration from these ideas, while also remembering that continuous systems enable additional capabilities[3] due to conservation laws and continuity.

Learning the Environment and Synthetic Ids

The state concept implies a separable categorization of the *world* into groups representing focus points based on characteristics of importance; We'll call this "a state" in the considered *environment* representation, and can define the state of any entity of interest by its exact "world" parameter configuration (see fig. 1).

Consider first a simple grid representable problem – to bring the situated entity to one particular parameter configuration. Grid coding(Sutton & Barto, 1998) as illustrated in fig. 1 is a simple discretization scheme for this problem. The task could be seen as a *basal desire* for the agent, and the policy for this *basal desire* could be trained by off-policy Q-learning.

[3]Being continuous exposes certain characteristics and prohibits others, e.g. discrete jumps for parameters are not possible for physical systems, something that could be taken into account.

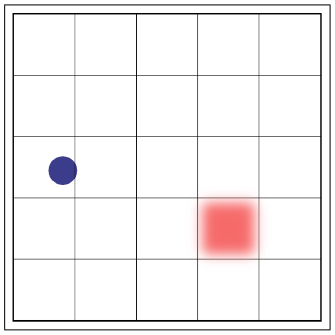
Figure 1: Example of $N5$ representation of a 2D (continuous) parameter space by grid coding, splitting each parameter into 5 discrete steps. The blue dot would have a state (1,3) in this parameter set; 25 different challenges could be defined in this representation, e.g. "go to position (4, 4)".

Id Priority

Let there be $K = N \times N$ such learning tasks, one for each state in this representation. Contrary impulses could then co-exist side by side, learning how to achieve their respective *obsessive state*. We will refer to this specific type of basal drive as "*Id*" referring to Freud's description of a distinct, yet interactive agents in the psychic apparatus(Freud, 1940), as defined in his structural model of the psyche. Since each *Id* is learning how to get to a particular state in one *environment representation*, we refer to this activity for the agent comprised of these *Id*s as *learning the environment*.

Let there further be a scalar weight W associated with each *Id*, representing how much a (super-)agent is "listening" to a particular *Id*'s opinion for the next action selection. All *Id*s with a positive W would contribute to pulling the agent toward their obsessive state, making the weighted sum represent a daemoncratic decision basis for the next action. We could think of this as a *Q-field* that represents the experience-based correlations formed over many *Ids*. At any moment, the learned Q-field represents the aggregated Q value in this environment representation, enabling action selection to be based on the input from a dynamic number of *Id*s according to the current attention scheme of the agent.

Experimental Setup

Development and testing of the presented theory has been done in a digital sand-box environment with certain real-world characteristics. PyGame's WaterWorld challenge (Tasfi, 2016) is a "game" with simple rules, but slightly convoluted mechanisms involving inertia. The task is to control a blue dot ("creep") in a continuous 2D plane such that green creeps are captured and red creeps avoided (see lower layer in fig. 2). When a creep

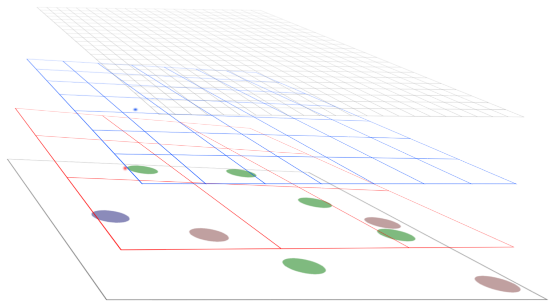

Figure 2: Illustration of Multi-Shepherd state representation / multiple discretization representations acting in parallel. [Red] N3 representation [Blue] N7 representation [Black] N23 representation. An R/B/B dot is marked for each representation's state corresponding to the agent's exact location.

is encountered, one receives a reward of $+1$ for green and -1 for red, before they reappear with a random location, speed, and demeanor(color). When there are no green instances left, the board is reset and a reward of $+5$ is received. 8 simultaneous creeps are used in for the experiments in this work, as shown in the bottom plane of fig. 2.

All results presented here comes from learning reactive control with 1-step lookahead for *Id* policies with no prior knowledge. The binary sparse nature of the reward signal/ score makes signal processing important before reults is presentable; The main experiment in this work is performed by performing 100 individual runs of the 4 considered agents. Experiments presented in fig. 3 is done with single runs because of long execution time for the higher resolutions. Cumulative sum is a robust signal processing approach and considered acceptable for illustrating how learning speed varies with the resolution of the environment representation ($N5, N10, \ldots$).

Unfortunately, no reports on RL agents that solve this problem have been found for comparison, and the task is considered to be a development platform and proof-of-concept toward real-world RL applications.

Results and Discussion

Our first verification of the effect of Id Shepherd learning on sample efficiency can be seen in fig. 3. We see that complex reactive behaviour (following or avoiding 8 simultaneous "creeps") is expressed by a positive gradient for cumulative score. Any result higher than (on average) 9 points (including completion reward) requires the agent to catch all green creeps for board reset. With creeps being respawned with a random color after every encounter, a total score of $1.000 - 4.000$ points during before 150.000 time steps of these agents' life is impressive for this problem complexity. Note that the last instance of a green creep must be caught among $[8 - 1 = 7]$ reds before every board reset and that all behavior is learned!

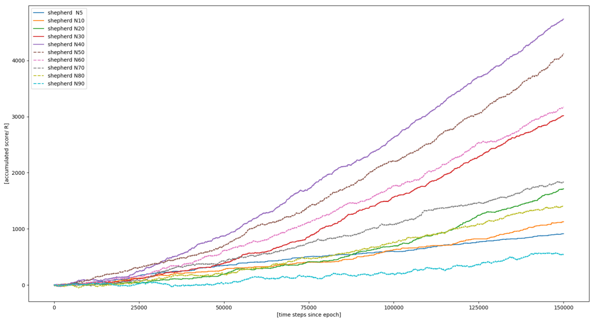

Figure 3: Accumulated Reward for runs with different shepherd resolutions, $N5 - N90$ over 150.000 time steps. Lower-resolution Shepherd quickly becomes proficient but learning goes into saturation (integral becomes linear – see $N5$). The shape of the $N50+$ curves indicates that there is still learning going for these representations in this time-frame.

For a first indication on interdimensional learning efficiency we can see the result of learning across multiple environment representations in fig. 4. The experiments are by agents designed like individual layers and the combination agent as presented in fig. 2. Curves illustrate the immediate proficiency of agent during the very first 50.000 time steps for each agent, represented as the per-timestep reward as a function of time. Agents start from scratch with no prior experience or hidden hyper-parameter settings; All aspects of these experiment are identical except for Shepherd composition, making this an interesting case for comparing a potential efficiency increase from the individual policies by composition.

By following a prioritized sum across multiple state representations, the agent actually demonstrates a higher proficiency for this task than an algebraic sum of separate runs of single-Shepherd agents. This is considered particularly impressive due to the non-linearity mentioned as the "9-point-barrier". Despite of this hypothesized favorization due to saturation mechanisms, the Multi-Shepherd learning agent actually learns faster, to a higher proficiency, than the sum of its components.

Concluding Remarks

This work shows us how a distributed design inspired by nature can help toward modularity in RL, prioritized aspect learning, and ability to alter the set of agent environment representation over time. The resulting agent

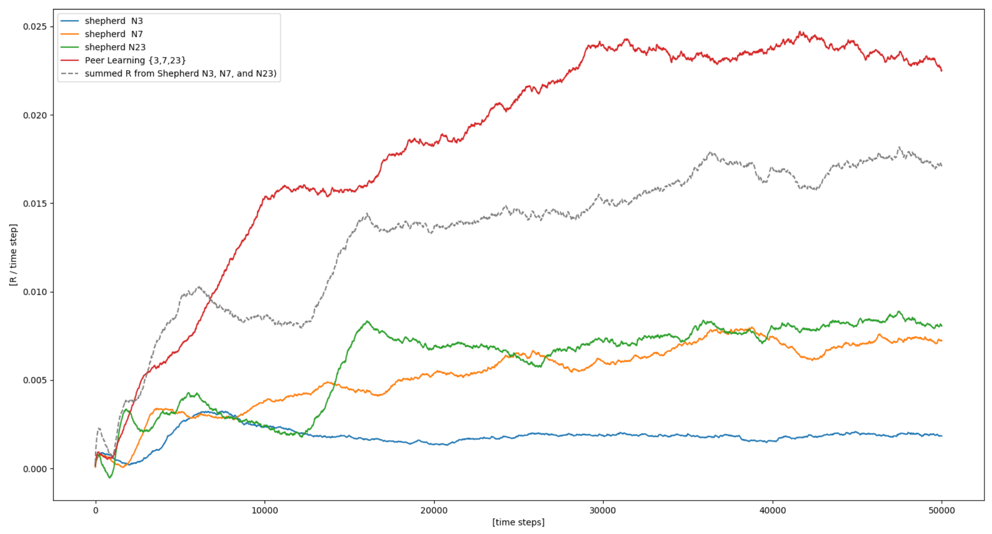

Figure 4: Proficiency of following policy generated by the individual Shepherds, and the joint policy constructed by the design in fig. 2. All lines are a mean over 100 individual runs for the respective experiment.

concept has been demonstrated for an environment with 2 continuous parameters with inertia, and the hypothesized real-time learning capabilities are illustrated for a simple agent over 3 prioritized sub-agents, consisting of 587 parallel learners, over different environment representations.

Our experiments show that by following in interdimensional policy, i.e. a combination of policies across state spaces, the agent achieved a higher proficiency at the task than the summed score of agents following policies generated by equivalent sub-components. The capability of dynamically extending the number of considered elements at any time is still considered to be the most important contribution of this work. These results imply that the agent can be adapted with a changing environment and over different parameter spaces, as well as that novel aspects of a task can be learned or introduced at any time.

References

Amarjyoti, S. (2017). Deep reinforcement learning for robotic manipulation - the state of the art. *CoRR*, *abs/1701.08878*. Retrieved from `http://arxiv.org/abs/1701.08878`

Freud, S. (1940). An outline of psychoanalysis (vol. 23). *Standard Edition (J. Strachey)*, 198.

Moser, E. I., Kropff, E., & Moser, M.-B. (2008). Place cells, grid cells, and the brain's spatial representation system. *Annu. Rev. Neurosci.*, *31*, 69–89.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... others (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT press.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., & Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th international conference on autonomous agents and multiagent systems-volume 2* (pp. 761–768).

Tasfi, N. (2016). *Pygame learning environment.* https://github.com/ntasfi/PyGame-Learning-Environment. GitHub.

van Seijen, H., Fatemi, M., Romoff, J., & Laroche, R. (2016). Separation of concerns in reinforcement learning. *arXiv preprint arXiv:1612.05159*.

Van Seijen, H., Fatemi, M., Romoff, J., Laroche, R., Barnes, T., & Tsang, J. (2017). Hybrid reward architecture for reinforcement learning. In *Advances in neural information processing systems* (pp. 5392–5402).

Watkins, C. J. C. H. (1989). Learning from delayed rewards.