Expert based learning for planning behavior recovery

Mehdi Mounsif, Sébastien Lengagne, Benoit Thuilot, Lounis Adouane

15 mai 2018

1 Introduction

In this work, we focus on transfering prior knowledge from classic planning and control algorithms into neural networks. We validate our method on a 2d planar environment. In a second part, we open up our reflexion to reinforcement learning algorithms and their suitability with multi-robots systems.

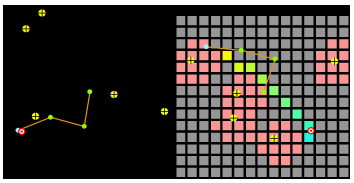

FIGURE 1 – A snapshot of the 2D environment. On the left, a raw environment. On the right, we environment is enhanced using the $\mathbb{A}^\star$expert

2 Methods

Various approaches are suitable for robot control. Reactive methods are fast, but can liable to fail, usually due to local minimas. On the other hand, planning and congnitive methods can provide safe and constraints satisfying path at the cost of heavy computation. In the following, we explain how we plan to leverage machine learning advances to provide a framework combining the best of both worlds.

2.1 Environment

Our environment consists of a planar robot, of which the number of degrees of freedom is controllable, a target and optional obstacles. It is possible to add any number of obstacles, either randomly placed within user-defined limits or manually. These control are also available for the target position parameters, which can prove useful for curriculum learning.

The environment also encompass various classes of experts for control. So far, three experts are defined : An IK controller based on the inversed jacobian, a potential field expert [1] and an $\mathbb{A}^\star$expert, [2]. In their current state, these experts are tuned to compute trajectories for particules rather than a full robot. Hence, some mandatory constraints of rigidbodies are, for now, overlooked by these experts.

2.2 Supervised Learning

We defined an environment with fixed obstacles and generate a set of labeled trajectories, using the $\mathbb{A}^\star$expert. For each timestep of the trajectory, we map the environment state to the joints deltas provided by the expert. The environment state concatenates the robot internal state i.e : joints angles, with for each obstacle, a vector between the obstacle center and the effector. Our incentive was to see whether we could recover some of the expert's behavior *via* ponctual perceptions. More specifically, we used a simple feedforward network, with two layers, with respectively 256 and 128 units, connected by ELU activation function, and a hyperbolic tangent for the output. In order to improve the state representation, we scaled up our system to use convolution filters at the beginning. Hence, in this case, the network maps a $64 \times 64 \times 3$ tensor to the joints velocities. However, in this configuration, results were disappointing, see Table 2 and Figure 2 for architecture details.

2.3 Results

The first tables focuses on the performances of a controller using a low-level state representation. For each table, we compare the case where the controller is trained over a fixed configuration of obstacles, only the target moves between each episode and the case where both the obstacles and the target are shuf-

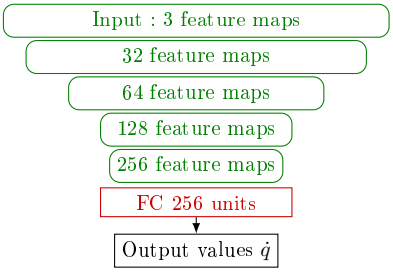

FIGURE 2 – Model for mapping an image to joint velocities, based on an expert behavior. Each rounded corners rectangle is a convolution layer.

fied. Each result is the average of 5 runs, with each run being 10000 episodes. The results using low-level representation show us that while this setting works with fixed obstacles, see Table 1, it doesn't seem possible to scale it up, i.e : recover a coherent behavior when the obstacles are liable to move between each run.

TABLE 1 – Low level representation. 3 obstacles

Joints	Success fixed	Success moving
3	87.8%	22.5%
5	76.2%	21.1%

TABLE 2 – Image representation. 3 obstacles

Joints	Success fixed	Success moving
3	35.3%	8.1%
5	31.3%	7.7%

3 Perspectives / Reinforcement Learning

To overcome the situation in which we fail to recover the expert behavior, see Table 2, we are currently investigating an enhanced architecture, see Figure 3. We hope that a memory module will provide the means to infer planning from an image, encoded to a smaller dimension for ease of learning, see [3]. In the future, if this pipeline proves successful, the $\mathbb{A}^\star$expert could be switched with a more complete method, allowing to retrieve licit movements.

Next, we plan to introduce reinforcement learning methods [6], [4]. Indeed, one of our strongest reasons

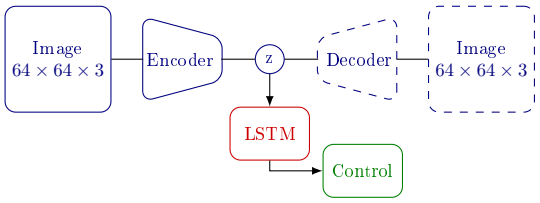

FIGURE 3 – Our model for recovering an expert behavior in a cluttered environment, inspired from [3]

to lean toward reinforcement learning has to do with extending and scaling up our system to various robots acting in cooperation. Behavior's quality in the supervised case mostly depends on the dataset's quality. However, designing a dataset with cooperative robotic controls proves difficult and would defy the very purpose of learning it. Instead, in the future, we plan to, given a robot controller, either obtained from the supervised learning process or an imitation learning process, improve their cooperative behavior based on RL and mutual interaction, see [7], [5].

Références

[1] Howie Choset, Kevin Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia Kavraki, and Sebastian Thrun. *Principles of Robot Motion : Theory, Algorithms, and Implementation*. 01 2005.

[2] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. 37 :28–29, 12 1972.

[3] D. Ha and J. Schmidhuber. World Models. *ArXiv e-prints*, March 2018.

[4] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *CoRR*, abs/1606.03476, 2016.

[5] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environnements. *arXiv :1706.02275v2*, June 2017.

[6] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, 2017.

[7] Bansal Trapit, Pachocki Jakub, Sidor Szymon, Sutskever Ilya, and Mordatch Igor. Emergent complexity via multi-agent competition. *arXiv - OpenAI Technical Report*, 2017.