International Journal of Control, Automation and Systems VV(X) (YYYY) 1-5
http://dx.doi.org/10.1007/s12555-xxx-xxxx-x
ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Coaching with PID Controllers: Accelerate Reinforcement Learning with PID Coaches in the Pendulum Simulations

Zongqiang Pang*, Liping Bai, and Yang Yang

Abstract: We propose a Proportional Integral Derivative (PID) controller-based coaching scheme to expedite reinforcement learning (RL). We demonstrate the feasibility and effectiveness of our approach in the inverted pendulum, inverted double pendulum simulation. Previous attempts to merge classical control with reinforcement learning focus on utilizing existing controllers as teachers to the RL agents, but they require high fidelity controllers and the benefit comes with an implicit cap on what is attenable by the RL agent. We ask if it is possible to accelerate RL with even a primitive hand-tuned PID controller and not inadvertently inflicting any limit to the RL agent. Instead of being teachers, the PID controllers function as coaches, whose job is not to provide templates to be imitated after but to be ready with necessary interventions. RL training can be expedited so long as the PID coaches can provide the appropriate assistance with a reasonable success rate. We validate our approach in the Mujoco Inverted Pendulum and Inverted Double Pendulum simulations. We conclude from the data that when the coaching structure between the PID controller and its respective RL agent is designed appropriately, the agent's training can be accelerated, yielding uncompromised training results. This is an important proof of concept that controller-based coaching can be a novel and effective paradigm for merging classical control with learning and warrants further investigations in this direction. All the code and data can be found at github/BaiLiping/Coaching

Keywords: Reinforcement Learning, Control, Learning for Dynamic Control, L4DC

1. INTRODUCTION

Learning for Dynamic Control is an emerging field of research located at the interaction between classic control and reinforcement learning (RL). Although RL community routinely generate jaw-dropping results that seem out of reach to the control community [1] [2] [3], the theories that undergird RL are as bleak as it was first introduced [4]. Today, those deficiencies can be easily papered over by the advent of Deep Neural Networks (DNN) and ever faster computational capacities. However, for RL to reach its full potential, existing control theories and strategies have to be part of that new combined formulation.

There are three ways that classic control finds its way into RL. First, theorists who are well versed in optimization techniques and mathematical formalism can provide systematic perspectives to RL and invent the much needed analytical tools [5] [6] [7] [8] [9]. Second, system identification researchers are exploring all possible configurations to combine existing system models with DNN and its variations [10] [11] [12] [13] [14]. Third, proven controllers can provide data on successful control trajectories to be used in imitation learning, reverse reinforcement learning, and "human"-in-the-loop learning [15] [16] [17] [18] [19].

Our approach is an extension of the third line of research. We propose a coaching structure between the classical controllers and the RL agents. Professional athletes don't get where they are via trial-and-error. Their skillsets are forged through painstakingly designed coaching techniques. At the top level, a coach's objective is not to be a template for the athletes to imitate after, but rather is to facilitate data collection on critical states.

Previous researches [20] [21] [22] are about making a functioning controller works better. They require high-quality controllers to begin with, and the improvements brought about by the RL agents are merely icing on the cake. In addition, the controllers inadvertently impose limits on what can be achieved by the RL agents. If, unfortunately, a bad controller is chosen, then the RL training process would be hindered rather than expedited.

Environment	PID Controller	Full Score	PID/Full Score
Inverted Pendulum	240	1000	24.0%
Double Pendulum	1107	10000	11.07%

Table 1. Performance of the PID controllers, as indicated by the average score over 10 episodes.

In our approach, the Coaches are PID controllers that can only score a fraction of what is available, as shown by Table 1. Yet, even with such barely-functioning controllers, when appropriately structured, RL agent training acceleration is still observed in our experiments, as shown by Table 2. "Coaching" means the PID controller takes over when the

Zongqiang Pang, Liping Bai, Yang Yang are with the College of Automation & College of Artificial Intelligence, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu,210000 China email:zqpang@njupt.edu.cn. * Corresponding author.

RL agent deviated from the essential states. Our approach also differs from previous researches in one significant way: interventions from the coach are hidden from the RL agents and they are not part of the training data. We also restrain from reward engineering, such that we can be confident that the observed acceleration does not stem from other alterations. The implementations would be detailed in subsequent sections.

Environment Name	Target Score	Measure	With PID Coach	Without Coach	Percentage Improvement
Inverted	800	5 Wins	90	160	43.8%
Pendulum		Average	91	159	42.8%
Double	7000	5 Wins	1024	1909	46.3%
Pendulum		Average	1391	2031	31.5%

Table 2. Comparison Between RL Agents Trained With and Without PID Coaches.

In section II, we present the idea of controller-based coaching. In section III, we present the results of our experiments in the inverted pendulum and inverted double pendulum simulations. We conclude what we have learned and layout directions for further research in section IV.

2. PID CONTROLLER BASED COACHING

Reinforcement Learning is the process of cycling between interaction with an environment and refinement of the understanding of that environment. RL agents methodically extract information from experiences, gradually bounding system models, policy distributions, or cost-to-go approximations to maximize the expected rewards along a trajectory. RL is an umbrella term that encompasses distinctive learning genres, which means updating parameters for the function approximators. A general taxonomy of RL is listed below:

Model-Based or Model-Free learning refers to whether or not learning is used to approximate the system dynamics function. If there is an explicit action policy, it is called on-policy learning. Otherwise, the optimal action would be implicitly captured by the Q value function, which would be called off-policy learning instead. Importance sampling allows "limited off-policy learning" capacity, which enables data reuse in a trusted region. Online learning means interleaving data collection and iterative network parameters update. Offline learning means the data is collected in bulk first, and then the network parameters are set with regression computation. Batch learning, as the name suggested, is in between online and offline learning. An agent would first generate data that fill its batch memory and then sample from the batch memories for iterative parameter update. Generate new data with the updated parameters. This taxonomy is somewhat outdated now. When Richard Sutton wrote his book [23], the algorisms he had in mind fall nicely into various categories, today, however, the popular algorisms would combine more than one route to derive superior performance and can't be pigeonholed.

A fundamental concept in RL is convergence through bootstrap: instead of asymptotically approaching a known target function, the bootstrap method converges to an assumed target first and then update the target assumption based on collected data. When evaluating estimation functions with belief rather than the real value, things could just run around in circles and never converge. Without any guidance, the RL agent would have explored all the possible states, potentially resulting in this unstable behavior.

How to collect data in the most efficient manner (Exploration Exploitation Trade-off) and avoid the instability mentioned above in the bootstrap process is a primary focus of researchers. One such method is to give more weight to critical states. Not all observational states are created equal. Some are vital, while others have nothing to do with the eventual objective. For instance, in the inverted pendulum task, any states outside of the Lyapunov stability bounds should be ignored since they would lead to an episode's eventual termination anyway.

There are statistical techniques to distinguish critical states from the non-essential ones, and imitation learning works by marking crucial states with demonstrations. However, the former approach is hard to implement, and the latter one requires high-quality controllers. Our proposed controller-based coaching method is easy to implement and does not have stringent requirements on the controllers it uses.

Controller-based coaching works by adjusting the RL agent's trajectory and avoid wasting valuable data collection cycle on the states that merely lead to eventual termination. When the agent is about to deviate from essential states, the controller will apply a force to nudge the agent back to where it should be, much like a human coach course-corrects on behalf of the athlete.

Crucially, the agent is oblivious to this intervention step, and it would not be part of the agent's data. Even if the controller didn't adjust the agent to where it should be, it would not have mattered since the RL agent is headed towards an episode's termination.

On the other hand, if the controller successfully adjusts the trajectory, the RL agent's next data collection cycle will be spent in a critical state. Therefore, all we require of the PID controller is a reasonable success rate in adjusting the RL agent's trajectory. We test our approach on four mujoco locomotion environments as a proof of concept, and in all four experiments, the hypothesized acceleration on RL training is observed.

3. EXPERIMENT SETUP

Mujoco physics engine [24], is one of many such simulation tools. We interface with it through a python wrapper provided by the OpenAI Gym [25]. We choose two environments for our experiments as a proof of concept: inverted pendulum, double inverted pendulum.

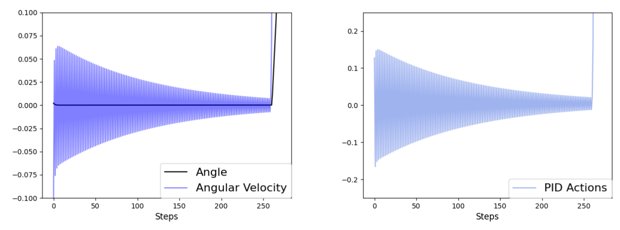

Fig. 1. Inverted Pendulum system controlled by the PID coach. The average score achieved by the PID controller is 240 out of 1000.

Every environment comes with a set of predetermined rewards and maximum episode steps. We did not tinker with those parameters. The only change we made to each environment is a controller-based coach ready to intervene when the agent steps out of the predetermined critical states.

We use tensorforce's [26] implementation of RL agents, specifically the Proximal Policy Optimization (PPO) agent because the learning curves generated by PPO agent are smoother, as shown by the spinning up [27] team. Our paper aims to indicate the controller-based coaching method's feasibility, and a smoother curve makes things easier.

In this paper, human judgment is the basis for the determination of critical states. In future works, we would like to provide a more systematic process for critical states' demarcation. We will provide our reasonings when we discuss our experiments in each environment.

Our experiments' code and data can be accessed via this depository. The folders are named after each environment, and in each folder, you will find a data record, an RL agent record, a agent.json file that indicates all the hyperparameters for the RL agent, a Normal file that trains a RL agent without a PID coach, a Coached file which trains a RL agent with a PID coach, a PID file which is the PID controller. In the PIDvsRL folder, you will find all data and codes that generate the plots shown in the following section.

3.1. Inverted Pendulum

The observation space of the inverted pendulum environment is: [Cart Position on X-axis, Cart Velocity, Pole Angle, Pole Angular Velocity]. The continuous action space is an action ranging from -1 to 1, where -1 means the actuator moves the cart to the left with maximum power and 1 means the actuator moves the cart to the right with full force. The maximum episode step number is 1000. The terminal state for the inverted pendulum is an angle of absolute value greater than 0.2 radius. The reward is 1 for each non-terminal step and 0 for the terminal step.

Figure1 shows how the PID controller manages the pole angle and its angular velocity. The PID parameters are the following: $k_p = 30, k_i = 0.01, k_d = 2.26$. While the PID controller tries hard to converge to zero, eventually, there would be too much accumulation on the x-axis, and the equilibrium breaks down.

Based on our observations of the system, we decide to put the boundary between critical and noncritical states on the angular velocity with an absolute value of 0.4. The RL agent is free to explore with the angular velocity with an absolute value smaller than 0.4, but once it goes above this bound, the PID controller will kick in, trying to decrease the velocity back to the 0.4 bound.

The experiment result is presented in Figure2. The black line indicates the RL agent trained without PID coach, and the blue line indicates the agent trained with a PID coach. The points on the line are the average score over 20 episodes, and the shaded area is the standard deviation. A predetermined standard is set at a score of 800. It takes the RL agent without PID coach 160 episodes to get five consecutive scores above the standard, and it takes the RL agent with PID coach 90 episodes to do the same. PID coach results a 43.8% acceleration in RL training. As measured by averaging over 20 episodes, it takes the RL agent without PID coach 159 episodes to go beyond 800, and it takes the RL agent with PID coach 91 episodes to do the same. PID coach results a 42.8% acceleration in RL training. Both RL agents trained with and without PID coach pass the evaluation, and their respective average scores are presented in the upper left corner. The final performance of the RL agent is shown in Figure3.

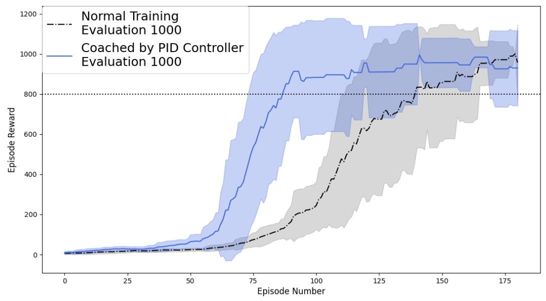

Fig. 2. Inverted Pendulum Experiment Result.

3.2. Inverted Double Pendulum

The inverted double pendulum has observation space of the following: [x position of the cart, sin(θ_1), sin(θ_2),cos(θ_1),cos(θ_2),velocity of x, angular velocity of θ_1, angular velocity of θ_2, constraint force on x, constraint force on θ_1, constraint force on θ_2]. θ_1 and θ_2 are the angles of the upper and lower pole respectively. The action space for Inverted Double Pendulum is an action ranging from -1 to 1, where -1 means the actuator moves the cart to the left with maximum power and 1 means the actuator moves the cart to the right with maximum power. A score

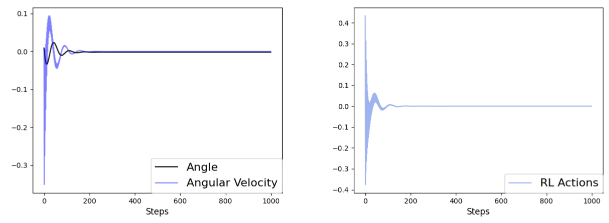

Fig. 3. Inverted Pendulum system controlled by the RL agent. The average score achieved by the RL agent is 1000 out of 1000.

of roughly 10 points is assigned to non-terminal states, based on the velocity on the x-axis. The detailed formula for score computation can be found at the OpenAI site.

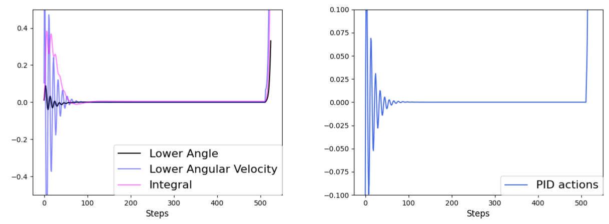

Fig. 4. Inverted Double Pendulum system controlled the PID coach. The average score achieved by the PID controller is 1107 out of 10000.

Figure4 shows how the PID controller manages the lower pole angle θ_1 and its angular velocity V_{θ_1}. The parameters for the PID coach is the following: $k_p = [-0.5, -0.5], k_i = [-0.04, -0.003], k_d = [-2.95, -0.56]$. The actuator action is computed with the following formula: action=$k_p[0] \times \theta_1 + k_i[0] \times \sum \theta_1 + k_d[0] \times V_{\theta_1} + k_p[1] \times \theta_2 + k_i[1] \times \sum \theta_2 + k_d[1] \times V_{\theta_2}$. The PID controller functions well until the equilibrium breaks down with too much disposition on the x-axis.

Based on our observation of the system, we decided to put the boundary between critical and noncritical states on the lower angle of with an absolute value of 0.2. The RL agent is free to explore with the lower angle with an absolute value smaller than 0.2, but once it goes above this bound, the PID controller will kick in, trying to nudge the lower angle back to the 0.2 bound.

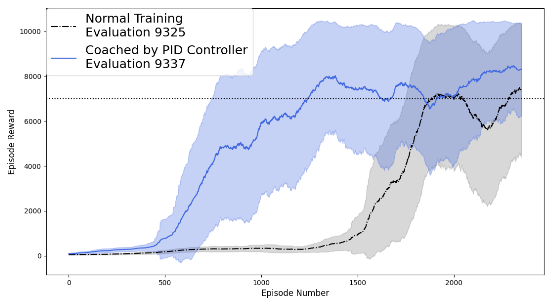

Fig. 5. Inverted Double Pendulum Coaching Result.

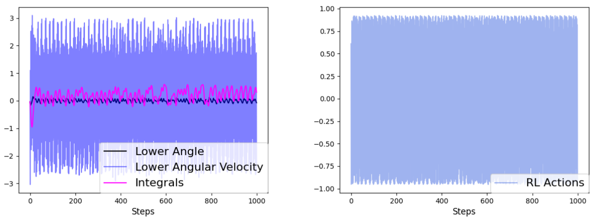

Fig. 6. Inverted Double Pendulum system controlled the RL agent. The average score achieved by the RL agent is 9319 out of 10000.

The experiment result is presented in Figure5. The black line indicates the RL agent trained without PID coach, and the blue line indicates the agent trained with a PID coach. The points on the line are the average score over 200 episodes, and the shaded area is the standard deviation. A predetermined standard is set at a score of 7000. It takes the RL agent without PID coach 1909 episodes to get five consecutive scores above the standard, and it takes the RL agent with PID coach 1024 episodes to do the same. PID coach results a 46.3% acceleration in RL training. As measured by averaging over 200 episodes, it takes RL agent without PID coach 2031 episodes to go beyond 7000, and it takes RL agent with PID coach 1391 episodes to do the same. PID coach results a 31.5% acceleration in RL training. The discrepancy between the two measurements is stems from the fact that RL agent trained with the PID controller has a higher variance on its performance. Both agents trained with and without PID coach pass the evaluation, and their respective average scores are presented in the upper left corner. The final performance of the RL agent is shown in Figure6.

4. CONCLUSION

This paper presents the PID controller-based coaching approach for accelerating RL training in pendulum environments. Unlike previous researches where the controllers function as teachers to the RL agents, our method emphasizes the coaching function of the controller. In both experiments, RL training is accelerated with the PID Coaches, for up to 46%. We believe our results warrant further investigation on coaching RL agents with classic controllers.

REFERENCES

[1] O. M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. W. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, pp. 20 – 3, 2020.

[2] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," *ArXiv*, vol. abs/1806.10293, 2018.

[3] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, 2020.

[4] D. Bertsekas and J. Tsitsiklis, "Neuro-dynamic programming," in *Encyclopedia of Machine Learning*, 1996.

[5] M. Han, L. Zhang, J. Wang, and W. Pan, "Actor-critic reinforcement learning for control with stability guarantee," *IEEE Robotics and Automation Letters*, vol. 5, pp. 6217–6224, 2020.

[6] E. Weinan, "A proposal on machine learning via dynamical systems," 2017.

[7] E. Dupont, A. Doucet, and Y. Teh, "Augmented neural odes," in *NeurIPS*, 2019.

[8] M. Betancourt, M. I. Jordan, and A. Wilson, "On symplectic optimization," *arXiv: Computation*, 2018.

[9] O. Nachum and B. Dai, "Reinforcement learning via fenchel-rockafellar duality," *ArXiv*, vol. abs/2001.01866, 2020.

[10] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," 2020.

[11] A. Mohan, N. Lubbers, D. Livescu, and M. Chertkov, "Embedding hard physical constraints in neural network coarse-graining of 3d turbulence." *arXiv: Computational Physics*, 2020.

[12] B. Lusch, J. N. Kutz, and S. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature Communications*, vol. 9, 2018.

[13] S. Bai, J. Z. Kolter, and V. Koltun, "Deep equilibrium models," *ArXiv*, vol. abs/1909.01377, 2019.

[14] F. de Avila Belbute-Peres, T. D. Economon, and J. Z. Kolter, "Combining differentiable pde solvers and graph neural networks for fluid flow prediction," *ArXiv*, vol. abs/2007.04439, 2020.

[15] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: the tamer framework," in *K-CAP '09*, 2009.

[16] ——, "Combining manual feedback with subsequent mdp reward signals for reinforcement learning," in *AAMAS*, 2010.

[17] X. Peng, P. Abbeel, S. Levine, and M. V. D. Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, pp. 143:1–143:14, 2018.

[18] X. Peng, E. Coumans, T. Zhang, T. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *ArXiv*, vol. abs/2004.00784, 2020.

[19] T. Paine, S. G. Colmenarejo, Z. Wang, S. Reed, Y. Aytar, T. Pfaff, M. W. Hoffman, G. Barth-Maron, S. Cabi, D. Budden, and N. D. Freitas, "One-shot high-fidelity imitation: Training large-scale deep nets with rl," *ArXiv*, vol. abs/1810.05017, 2018.

[20] L. Xie, S. Wang, S. Rosa, A. Markham, and A. Trigoni, "Learning with training wheels: Speeding up training with a simple controller for deep reinforcement learning," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6276–6283, 2018.

[21] I. Carlucho, M. D. Paula, S. A. Villar, and G. G. Acosta, "Incremental q-learning strategy for adaptive pid control of mobile robots," *Expert Syst. Appl.*, vol. 80, pp. 183–199, 2017.

[22] B. S. Pavse, F. Torabi, J. P. Hanna, G. Warnell, and P. Stone, "Ridm: Reinforced inverse dynamics modeling for learning from a single observed demonstration," *IEEE Robotics and Automation Letters*, vol. 5, pp. 6262–6269, 2020.

[23] R. Sutton and A. Barto, "Introduction to reinforcement learning," 1998.

[24] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[25] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *ArXiv*, vol. abs/1606.01540, 2016.

[26] A. Kuhnle, M. Schaarschmidt, and K. Fricke, "Tensorforce: a tensorflow library for applied reinforcement learning," Web page, 2017. [Online]. Available: https://github.com/tensorforce/tensorforce

[27] J. Achiam, "Spinning Up in Deep Reinforcement Learning," 2018.