[10.1.2021]

[Dynamics Randomization Revisited: A Case Study for Quadrupedal Locomotion]

Summary

This paper surveys about the dynamics randomization in the sim-to-real application for robots and concludes that dynamics randomization is not necessary and also sufficient if some other design parameters is not serious chosen.

Some key points of this paper:

- Not necessary: for some parameters, such as mass, proportional gain, latency → in simulation, just performs a simple first-in-first-out buffer to model latency and in the physical environments, the latency is naturally existing, slope, etc. The solution is that dynamics randomization can lead to conservative policies that compromise the performance while gaining extra robustness.

- Not sufficient if some parameters are not well chosen. In this paper the authors research on the if the velocity information is included in the state (fro RL) and the choice of proportional gain.
    - High proportional gain leads to a position control → accurate, but dangerous when contact exist a lot
    - low proportional gain leads to behavior like a torque control → not accurate, but robust.

Solution is that dynamics randomization matters if for example, latency of the real experiment is too much (over 17 ms in this paper's robot), then randomization matters.

Key take-away from the paper is that dynamics randomization should be conducted on those parameters that do matter….. otherwise the overall performance will be compromised.

Major Analysis and Comparison

1) This paper uses a strategy to smooth the target joint position → a low path filter to average the desired joint with previous target joint → not natural to me

    - Adding MPC to track can ease this issue, adding similar cost into network, would it help?
        - Can this effect somehow be translated/associated between MPC and network. i.e. Integrate MPC into neural network?

2) Applied on the real robot for the first time

Thoughts

- What is dynamics randomization?
    - In the training phase, the parameter of the of the simulation system are randomization (bounded in some range) in order to let the learned model be robust.
- In what case it is useful, in what case it is not useful?
    - See parameter that changes dramatically from sim to real, then randomization is useful.
    - Some parameters that we actually don't need to randomization and it also works. With randomization, the performance is compromised because of the extra difficult added to train the network.
    - For example, mass randomization doesn't help that much, does it mean that the network is able to learn the "linear" relation between the mass and the target joint? (for example, q = cM, if c is learned, then q changes with M changes → there is no need to randomize M anymore)
- Does ANYmal learning to walk paper use dynamics randomization?
    - Yes, in that paper it uses this. But in some other papers, without using dynamics randomization achieves better results → that's why this paper is written to study that conflict.
- What kind of parameters are actually used in the simulation, are they physical parameter that has actual meaning?
    - For example, masses and inertial moments of the body link, some parameters that govern the actuator, control latency, etc.
- Do you think it makes sense that use MPC to further track the target joint position that is computed by the neural network?
    - This does not makes sense to me, but if we can somehow add MPC into the network framework,……xxxxxxxx not makes sense!! MPC itslef is providing joint target and this is further tracked by a PD controller. → to give torque.
    - The quality of target joint position computed by network, is it smooth? → if not, can use MPC.

- No one has done it yet seems like. Doesn’t make sense? No idea