[3.1.2021]

[Learning quadrupedal locomotion over challenging terrain]

Summary

This paper is a great milestone in combining learning and real robotics walking.

- Assumption/main idea of the paper: the proprioceptive signals can fully/partially

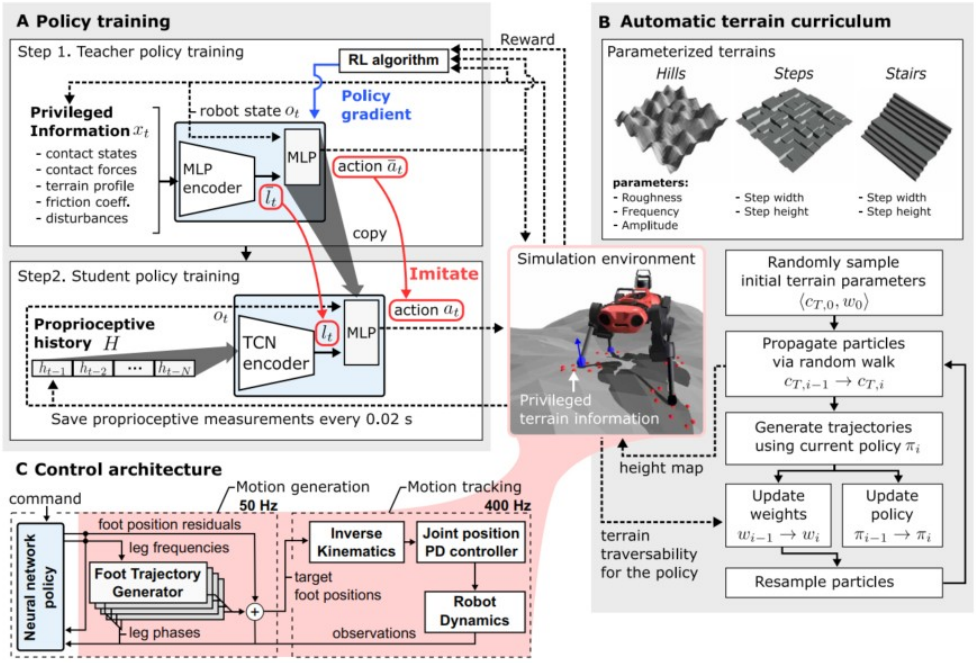
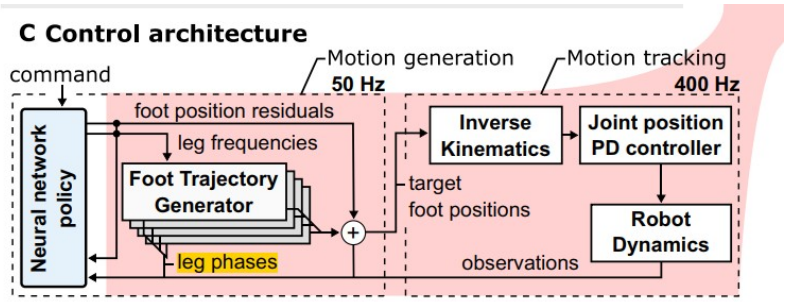


Fig. 4. Creating a proprioceptive locomotion controller. (**A**) Two-stage training process. First, a teacher policy is trained using RL in simulation. It has access to privileged information that is not available in the real world. Next, a proprioceptive student policy learns by imitating the teacher. The student policy acts on a stream of proprioceptive sensory input and does not use privileged information. (**B**) An adaptive terrain curriculum synthesizes terrains at an appropriate level of difficulty during the course of training. Particle filtering was used to maintain a distribution of terrain parameters that are challenging but traversable by the policy. (**C**) Architecture of the locomotion controller. The learned proprioceptive policy modulates motion primitives via kinematic residuals. An empirical model of the joint position PD controller facilitates deployment on physical machines.

reconstruct the unknown information → such as contact states, contact forces,

terrain profile, friction coefficients and disturbances → those are hard to get in real experiments, and we believe can be reconstructed/predicted simply by how robots senses the environments (robot sensor signals)

- Robots can still gain information, either in simulation or in experiments. In simulation, the robot observed states are command (target direction), orientation, base twist, joint position/velocity, leg phase indicator $\phi_i \in (0, 2\pi)$ for each leg and previous foot position.
    - In experiments, observation data is replaced with all observable data (take whatever you can get from the robot, including some estimators that estimates the orientation, position of the base, including history proprioceptive signals)
    - $o_t = h_{t:1:N} + f_O + joint\ history + previous\ foot\ position\ targets$.
    - ~~One thing I do not understand: $h_t \rightarrow x_t$, $o_t \rightarrow o_t(¿)$, but there is overlap in h_t and o_t for simulation (student learning).~~
        - The above statement is wrong: the assumption should be: *In the paper it says a key hypothesis is that the latent feature I can be recovered from a time series of proprioceptive observations, and this latent feature is some combination of robot state, terrain information, and the robot observations (estimations for something)*
    - The action space of the policy network is consisting of a 16 dim vector generate the trajectory through trajectory generator
- Terrain is updated during training, following the rule that it is neither too difficult for robots to traverse, not it remains the same all the time. Some strategies are used to update the terrain. → can refer to paper in the future, but not now.
- Another interesting part, check another paper PMTG
    - Inputs: command vector (direction) and a sequence of proprioceptive signals including base velocity, orientation, *joint states*. (I am confused now, need to check what information these proprioceptive contains on earth.)



- Outputs: joint position targets → that can be further tracked by a simple PD controller.
- Above is the total input output. For controller architecture alone, the input is leg frequency and foothold residuals, leg frequency goes to trajectory generator and will have part of the foot positions, plus the residuals we will get the desired foot position. Leg phases are extremely important. (see TG paper)
-

Comparison

- Comparison 1
- Comparison 2

Thoughts

- Some common questions when it comes to apply learning techniques into robotics:
  - Where to where learning → Input-output declaration, the output can be the direct torque signals that needed the motor/actuator to give → check another paper that “learn the motor behavior”. ➔ can also output desired trajectories (desired foot position and use inverse kinematics to compute the joint angle), let the torque controller do the rest part of the job.

        - I personally prefer the second idea, if we lean torques directly, that means we give up the dynamic information (or simply believe that the learned model can learn the model accurately as well, but there is no need risking this).
    - A difficult situation is that it is almost impossible to collect a lot of data from real experiments due to the fact the robot cannot really risk in the real life. An idea is to learn from simulation, where all data/information can be retrieved (terrain knowledge, etc.) → teacher, student learning techniques. Transfer learning.