Reinforcement Learning & PyCIGAR Architecture

- ***Reinforcement Learning:***
  - Branch of Artificial Intelligence which has demonstrated the ability to find optimal control policies in systems with complex dynamical interactions via intelligent simulation (i.e., “smart” trial and error)
- ***PyCIGAR:***
  - Python-based simulation framework used for training and evaluation of RL agents
  - Brings together:
    - Dynamic models of smart inverter functions (based on IEEE 1547)
    - Distribution system power flow tools (OpenDSS & custom solvers)
    - Open source reinforcement learning library (RLLib: https://docs.ray.io/en/master/rllib.html)

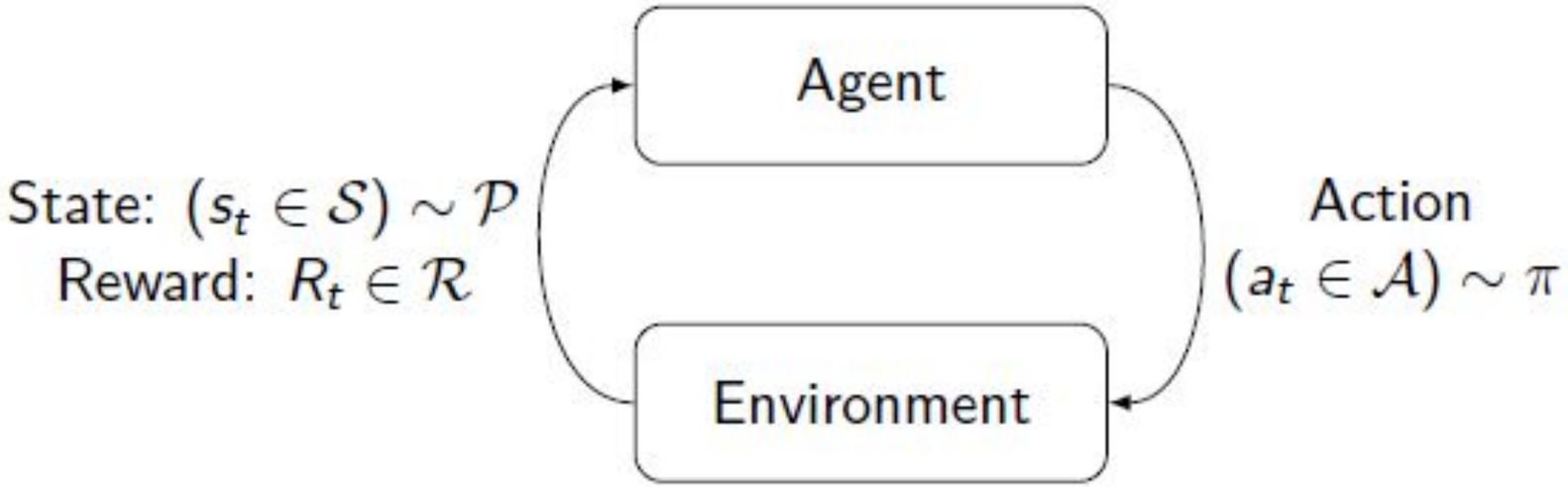
Markov Decision Processes (MDPs)

- *MDPs* provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision maker.
- Consists of the 4-tuple: $\left(S, A, P_a, R_a\right)$
- ***S*** is the set of possible states (i.e., the *state space)*
- ***A*** is the set of admissible actions
- $\boldsymbol{P_a}$ is the probability that action ***a*** in state ***s*** at time ***t*** will lead to ***s'*** at ***t+1***, or

$$P_a(s, s') = \mathcal{P}r(s_{t+1} = s' | s_t = s, a_t = a)$$

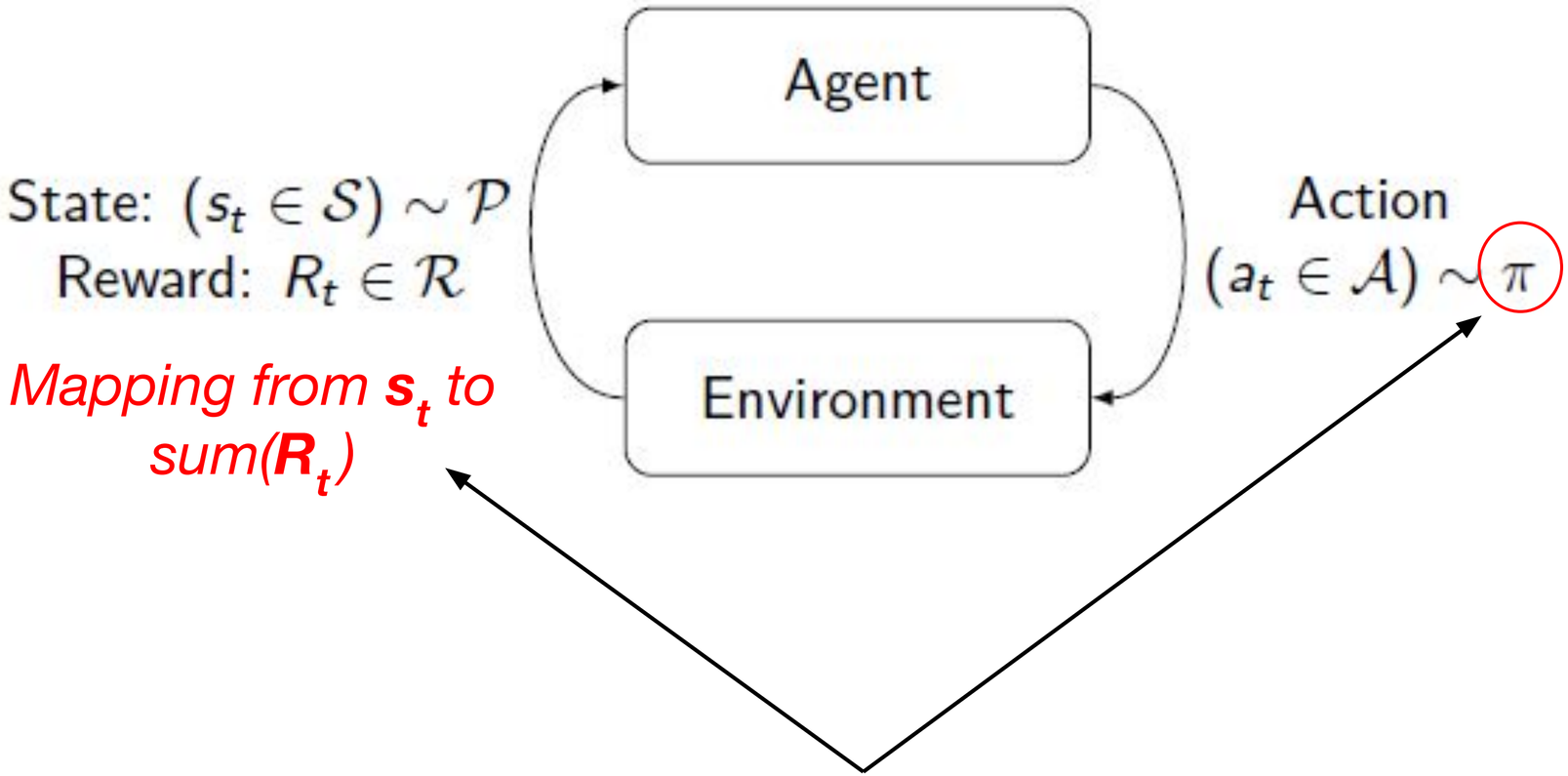
- $\boldsymbol{R_a(s,s')}$ is the reward received after applying ***a*** at state ***s*** to transition to ***s'***
- Goal of the optimization is to maximize the cumulative reward over a time horizon
- Can also be solved by dynamic programming or Monte Carlo techniques

Reinforcement Learning Agent Training

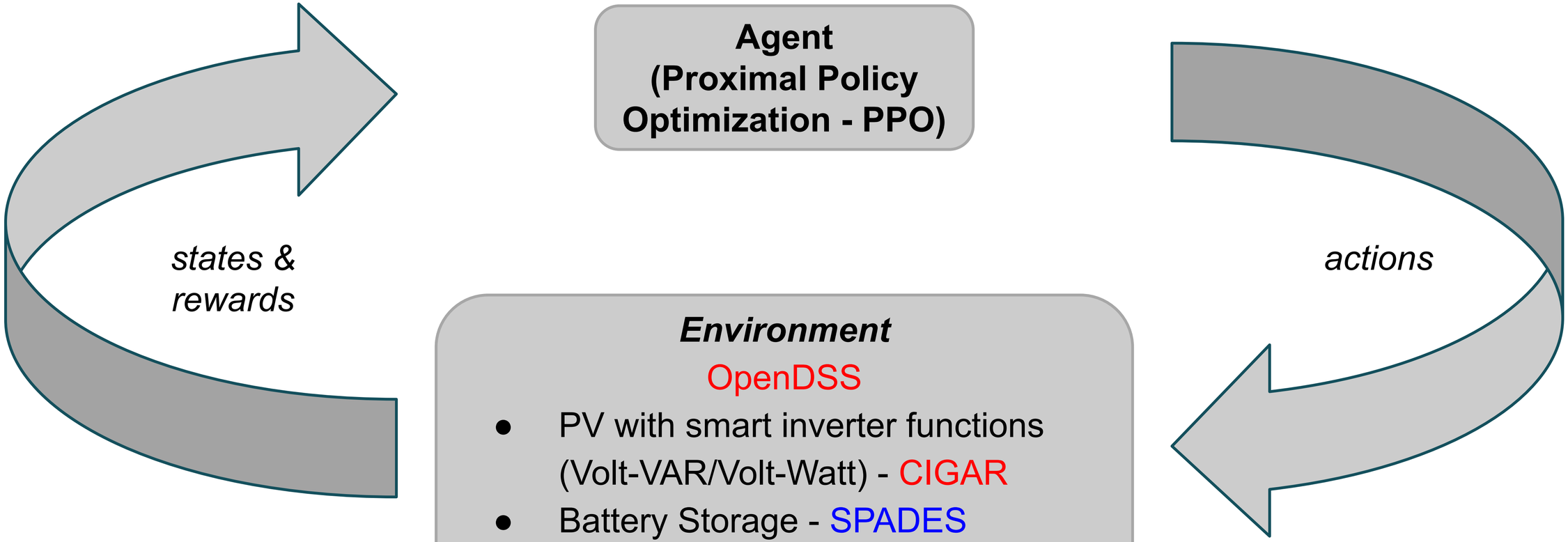


- At time **t** the agent selects action $\mathbf{a_t}$ from a policy **π** and applies that action to the environment
- Action causes a transition to a new state $\mathbf{s_t}$ which has an associated reward $\mathbf{R_t}$
- ($\mathbf{s_t}$, $\mathbf{a_t}$, $\mathbf{R_t}$) are used to update the policy **π**

Reinforcement Learning Agent Training

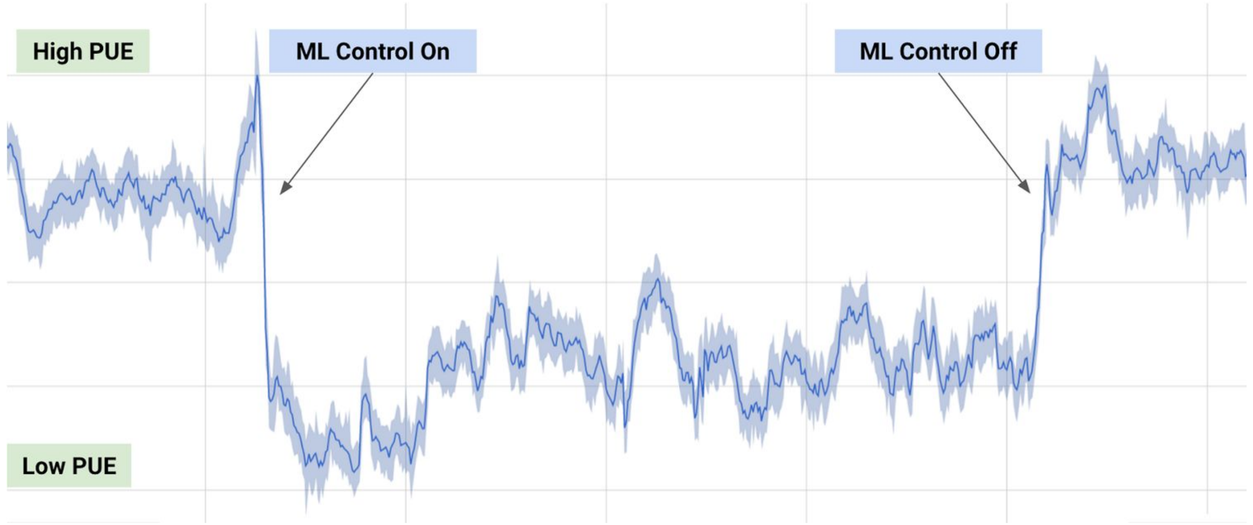


Reinforcement Learning Training Loop



Examples of Reinforcement Learning

Machine Learning Improves Google Data Center Cooling Efficiency

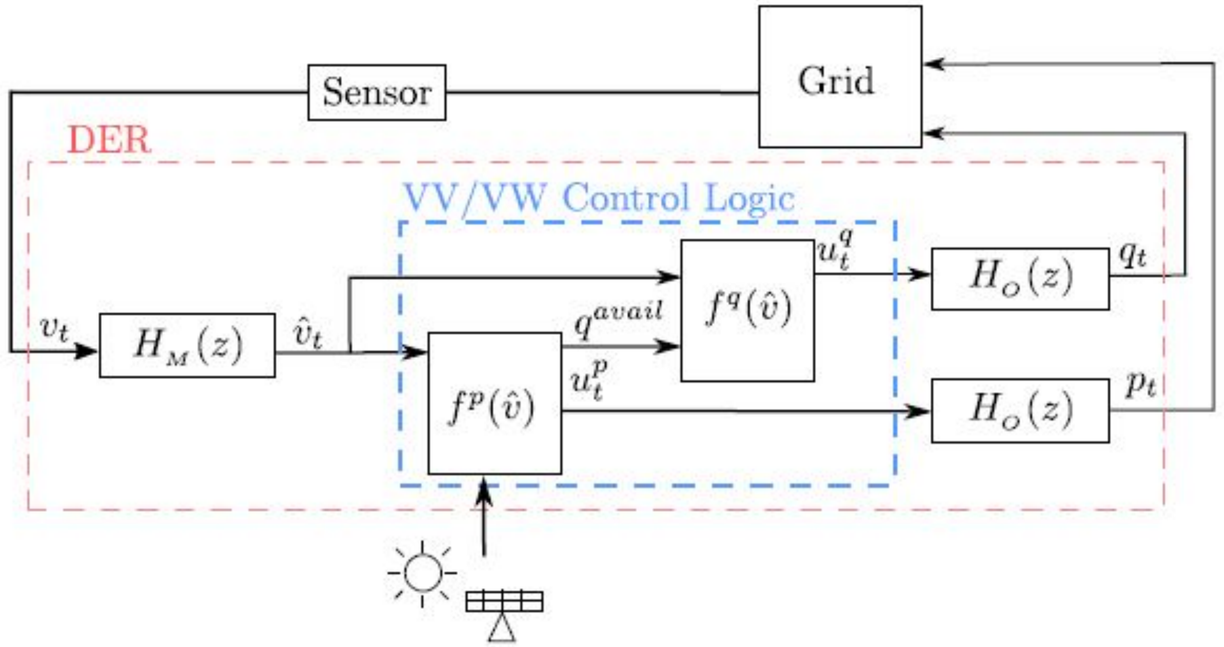


Reinforcement Learning Beats "Go" Expert

These algorithms enable decision-making in high dimensional uncertain problems

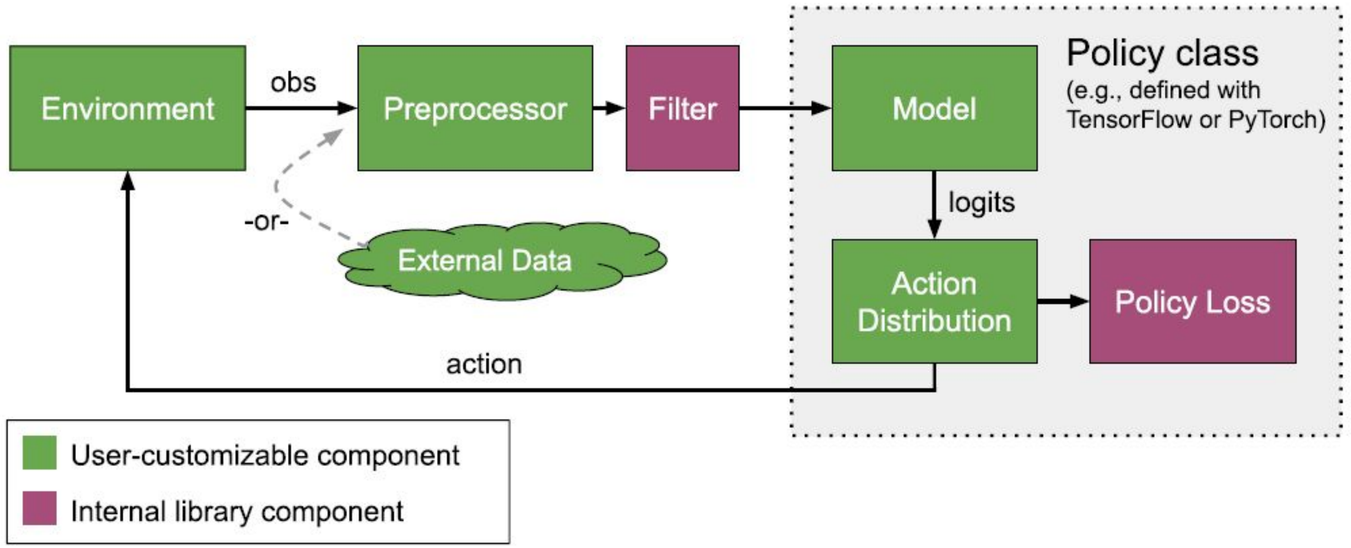
PyCIGAR - Smart Inverter Models

- PyCIGAR smart inverter module replicates the dynamic behavior of the Volt-VAR and Volt-Watt control
- $H_M(z), H_O(z)$ are low pass filters
- $f^q(\hat{v})$: inverter Volt-VAR curve
- $f^p(\hat{v})$: inverter Volt-Watt curve
- Volt-Watt precedence: available reactive power determined by excess inverter capacity

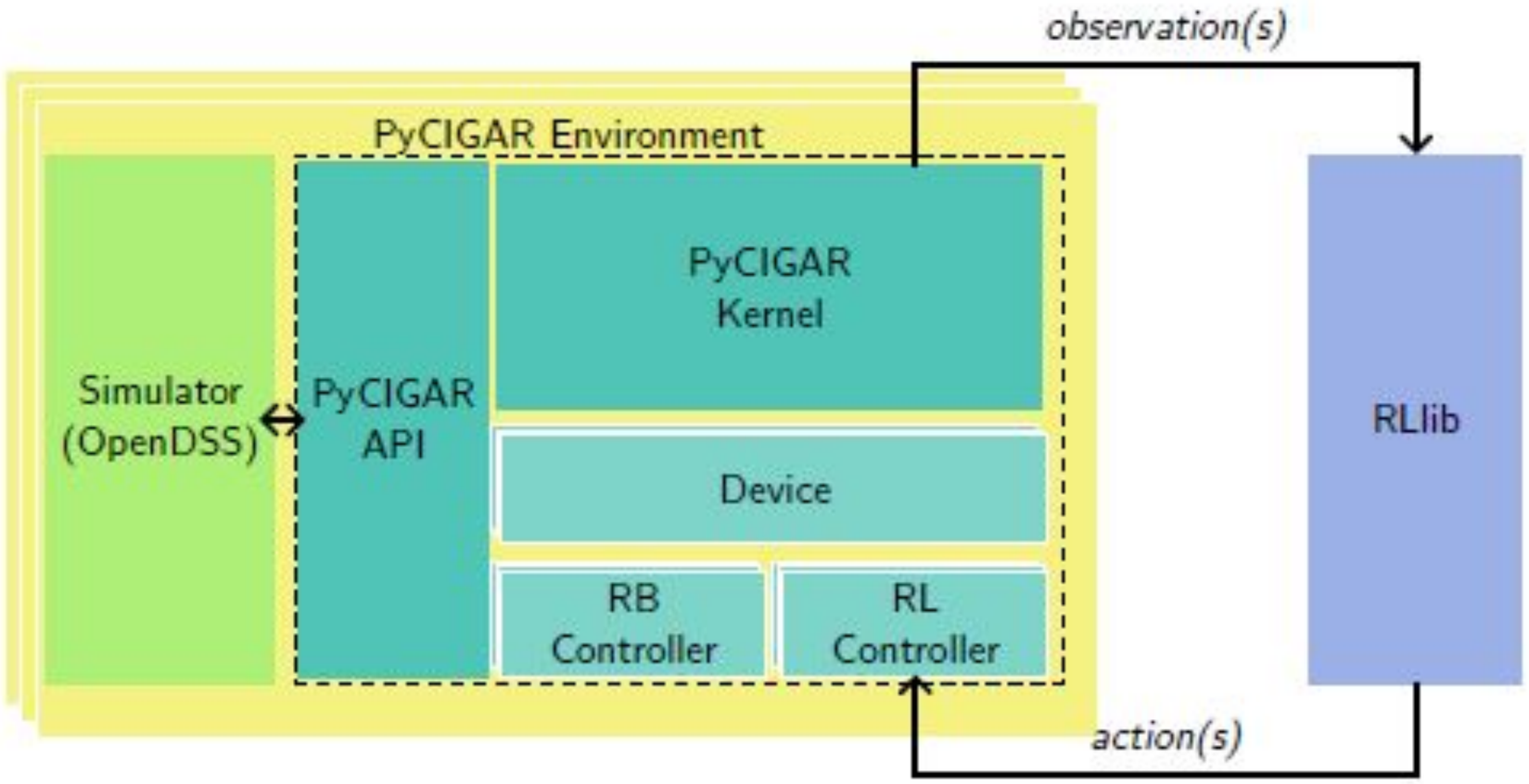


PyCIGAR - RLLib/Ray

- RLlib is a scalable open-source library for reinforcement learning
- Provides application support
  - Multi-agent/hierarchical
- Abstractions of multiple RL algorithms:
  - PPO, DDPG, A3C, etc.
- Supports distributed training
- Highly customizable



PyCIGAR Environment



*RB Controller: Rule-based controller
*RL Controller: Reinforcement learning controller

Questions/Discussion