Learning complex manipulation tasks by playing.

Problem

- Starting from a random initialisation, learn to perform manipulation tasks on the Human Support Robot (HSR).
- We formulate it as a reinforcement learning (RL) problem with sparse reward.

Simulation Environment

- Code: https://github.com/ascane/gym-gazebo-hsr

Key idea

- High-level scheduling of auxiliary tasks and the execution of auxiliary policies to explore efficiently ([1]).

Learning the policy (Actor $\theta$)

- The action-value function $Q_T(s_t, a_t)$ for task $T$
  \[ Q_T(s_t, a_t) = r_T(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}} \left[ \sum_{i=0}^{\infty} \gamma^i r_T(s_{t+i}, a_{t+i}) \right] \]
  where $T \in A \cup \{M\}$, $\pi_T = \pi_0(a|x, T)$.
- To learn the parameters, we optimise
  \[ L(\theta) = L(\theta; A) + \sum_{k=1}^{\lvert A \rvert} L(\theta; A_k) \]
  with $L(\theta; A_k) = \sum_{s \in A_k} \mathbb{E}_{s \in [S]} \left[ Q_T(s, a) \sim \pi_0(\cdot | s, T) \right]$. 

Learning the Q-function (Critic $\phi$)

- Since the policy parameters are constantly being updated, the trajectories are generated by different behaviour policies.
- The off-policy evaluation Retrace [2] is used to optimise the estimator $Q_T^P(s, a; \phi)$

Learning the scheduler

- To determine the current intention of the agent based on previous intentions.
  \[ R_{m}(T_{0:k-1}) = \sum_{m} \sum_{k} \gamma^k r_{m}(s_k, a_k) \]
  \[ \pi_S(a_k | s_k, T_{0:k-1}) = \sum_{m} \pi_0(a_k | s_k, T) P_S(T | T_{0:k-1}) \]
  \[ L(S) = \mathbb{E}_{P_S} \left[ R_{m}(T_{0:k-1} | T_{0:k-1}) \right] \]

Fig. 1: Schematic of an actor-critic agent.

Fig. 2: Network architectures for actor and critic taken from [1].

Fig. 3: Stacking boxes and the assembly challenge in simulation.

Fig. 4: The desired end configuration of the Siemens assembly challenge. Credits: Siemens Corporate Technology.

Fig. 5: HSR falls over / reaches forward too much / gets stuck under the table / runs away from the table.

Experiments

Stacking two boxes
- Stack the green box on top of the red one
- Three auxiliary task with sparse reward – Reach, Move, Lift.

Siemens Assembly Challenge
- Assemble different components to the end configuration as shown in Fig. 4.

Why is it challenging?
- Gazebo is too slow for RL algorithms.
- Hard to design auxiliary tasks for more complex tasks.