SPQR
RoboCup 2014 Standard Platform League

Dip. di Ingegneria Informatica Automatica e Gestionale
Sapienza Università di Roma, Italy

The RoboCup SPQR team participates to the *Open Challenges* introducing a reinforcement learning technique to improve the timing behaviour for a goalkeeper robot. The goal is to increase the number of saved balls by optimizing its dive times. As any reinforcement learning formalization, the agent iteratively learns an optimal policy $\pi^*(\theta)$ which maps every possible state in a consequent action and adjusts its training parameters evaluating the gained rewards.

A more formal definition of the learning process can be express as follows:

- **Task:** the goal is to produce an optimal keeper that is able to chose the most convenient time to perform an action. Each time the learner detects a potential scoring ball it evaluates the time interval the ball takes to reach the goal line. This interval is estimated in terms of ball velocity and trajectory, and then compared with a reference time provided by a supervisor.
- **Experience:** We run the algorithm in a virtual environment, including two playing robots and an external human supervisor, where the striker robot tries to score in the agent's goal. The learning session goes through several *episodes* (i.e. a goal attempt resolving either in a goal or in a saved ball). After each episode the supervisor looks at the outcome and assigns a negative or positive reward to the learner. After that, it moves the ball in a random position within the field.
- **Performance:** the performance measure is proportional to the positive and negative rewards during the process, i.e. the proportion of saved balls.

To this end we implemented and compared two different learning algorithms: one based on policy gradient techniques and another one on genetic algorithms. The virtual environment where experiments are carried out is part of the B-Human architecture (*https://www.b-human.de*), which features a RoboCup-dedicated simulation platform (SimRobot).

Policy gradient The algorithm aims at estimating the optimal parameters configuration for the policy $\pi = \{\theta_1, \ldots, \theta_N\}$ according to an objective function $F(\theta)$. At each episode the algorithm perturbs the current policy in accordance with its hyper-parameter ϵ and computes the overall gradient to guide the search in the *parameters space*.

Genetic algorithm The learner runs the GA algorithm to compute the optimal configuration θ^* for the objective function $F(\theta)$. After each session the agent calculates the time interval according to previously obtained θ and starts a new training episode exploiting the new generated population.

During the *3-minutes demo* we will show the results we obtained by using these algorithms. Since the training process is offline and needs a human support, we are going to set up a few diving behaviours showing the policy evolution in the parameters space (from the starting configuration to the optimal configuration θ^* computed offline).