Reinforcement Learning (RL) algorithms provide a way to optimally solve decision and control problems of dynamic systems. An RL agent interacts with the system by measuring the states and applying actions according to a certain policy. After applying an action, the RL agent receives a scalar reward signal related to the immediate performance of the agent. The goal is to find an optimal policy, which maximizes the long-term cumulative reward.

Define an n-dimensional state space $\mathcal{X} \subset \mathbb{R}^n$, and m-dimensional action space $\mathcal{U} \subset \mathbb{R}^m$. The model is described by the state transition function $x_{k+1} = f(x_k, u_k)$, with $x_k, x_{k+1} \in \mathcal{X}$ and $u_k \in \mathcal{U}$. x_k therefore represents the focus level of the human at time k. The reward function assigns a scalar reward $r_k \in \mathbb{R}$ to the state transition from x_k to x_{k+1}:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ r_{k+1} &= r(x_k, u_k) \end{aligned} \tag{1}$$

In this work a model-free setting was used, so x_{k+1} and r_{k+1} are obtained by the interaction with human. x_k is taken in a moment t, x_{k+1} is obtained after 15 (s) of interaction with a brain. Reward r_{k+1} is computed as a difference between x_k and x_{k+1}.

Define a finite set of discrete control input values $U = \left\{u^1, u^2, \ldots, u^N\right\}$ drawn from $\mathcal{U}$. In this work the set U represents stimuli. The Q-function can be updated by solving the Bellman equation:

$$Q_{k+1}(x_k)(u_k) = Q_k(x_k, u_k) + \alpha_k \left[r_{k+1} + \gamma \max_{u' \in U} Q_k(x_{k+1}, u') - Q_k(x_k, u_k) \right] \tag{2}$$

where γ is the discount factor (a user-defined parameter), α_k is the learning rate function, usually defined as $1/k$, x_k stands for the focus level in time k and u_k represents the applied stimulus at the same time. We working with continuous-valued waves, so the approximation should be used. There are several methods to approximate the Q-function for continuous state spaces. In this paper, we use the fuzzy Q-learning algorithm as it is guaranteed to converge and the fuzzy approximator allows us to interpret the values at each fuzzy set core directly as the Q-function value. The policy is defined by the following mapping:

$$h : \mathcal{X} \to \mathcal{U} \tag{3}$$

The most straightforward way to derive a policy corresponding to the approximate value function $Q(x, u)$ is:

$$h(x) \in \arg\max_{u \in U} \ \gamma Q(x, u) \tag{4}$$