Advanced Robotics

ISSN: 0169-1864 (Print) 1568-5535 (Online) Journal homepage: https://www.tandfonline.com/loi/tadr20

Multiple mini-robots navigation using a collaborative multiagent reinforcement learning framework

Piyabhum Chaysri, Konstantinos Blekas & Kostas Vlachos

To cite this article: Piyabhum Chaysri, Konstantinos Blekas & Kostas Vlachos (2020): Multiple mini-robots navigation using a collaborative multiagent reinforcement learning framework, Advanced Robotics

To link to this article: https://doi.org/10.1080/01691864.2020.1757507

ADVANCED ROBOTICS
https://doi.org/10.1080/01691864.2020.1757507

FULL PAPER

Multiple mini-robots navigation using a collaborative multiagent reinforcement learning framework

Piyabhum Chaysri, Konstantinos Blekas and Kostas Vlachos

Department of Computer Science and Engineering, University of Ioannina, Ioannina, Greece

ABSTRACT

In this work we investigate the use of a reinforcement learning (RL) framework for the autonomous navigation of a group of mini-robots in a multi-agent collaborative environment. Each mini-robot is driven by inertial forces provided by two vibration motors that are controlled by a simple and efficient low-level speed controller. The action of the RL agent is the direction of each mini-robot, and it is based on the position of each mini-robot, the distance between them and the sign of the distance gradient between each mini-robot and the nearest one. Each mini-robot is considered a moving obstacle that must be avoided by the others. We propose suitable state space and reward function that result in an efficient collaborative RL framework. The classical and the double Q-learning algorithms are employed, where the latter is considered to learn optimal policies of mini-robots that offers more stable and reliable learning process. A simulation environment is created, using the ROS framework, that include a group of four mini-robots. The dynamic model of each mini-robot and of the vibration motors is also included. Several application scenarios are simulated and the results are presented to demonstrate the performance of the proposed approach.

ARTICLE HISTORY
Received 14 October 2019
Revised 8 February 2020
Accepted 1 April 2020

KEYWORDS
Reinforcement learning; multi-agents; mini-robots; autonomous navigation; moving obstacles avoidance

1. Introduction

Recently, the design and realization of micro-manipulators and micro/mini-robots has become an important field of research. Potential areas of application are microsurgery, micro-manufacturing, micro-assembly, biomechanical design, and neurobiological control [1–3]. Several micro-actuation techniques have been developed, usually based on smart materials such as piezo-electric actuators, shape memory alloys, etc. The most popular micro-positioning motion mechanism is the stick-slip principle, [4], which is implemented using piezoelectric actuators. This principle is employed by the MINIMAN micro-robot presented in [5] and others. These platforms are capable of positioning accuracy of less than 200 nm and provide velocities of up to a few mm/s. The impact drive principle, a variant of stick-slip principle, is employed by the 3DOF micro-robotic platform Avalon, which provides step size of about 3.0 m and speeds up to 1 mm/s [6]. A different motion mechanism based on piezo-tubes is utilized by the Nano Walker micro-robot [7]. The first prototypes of this micro-robot were capable for minimum steps of the order of 30 nm and demonstrated a maximum displacement rate of 200 mm/s. The MiCRoN micro-robotic platform is equipped with piezoelectric actuators, with an integrated micro-manipulator [8]. The centralized control architecture of MiCRoN is presented in [9]. Kilobot, a low-cost robot designed for testing and validating algorithms for a swarm of robots, is presented in [10]. AMiRo, a modular robot platform that can be easily extended and customized in hardware and software is presented in [11]. Although the desired positioning accuracy can be achieved by using piezoelectric actuators, they need expensive and cumbersome power units that do not easily allow for untethered operation. An alternative drive mechanism is proposed by Vartholomeos and Papadopoulos in [12]. They presented a low-cost autonomous mini-robot (with dimensions of a few centimeters) driven by two DC vibrating motors that is able to translate and rotate, with micrometer positioning accuracy and velocities up to 1.5 mm/s. Positioning methodologies that compensate for the nonholonomic constraints of the same mini-robot are proposed in [13].

Reinforcement Learning (RL) aims at controling an autonomous agent in unknown stochastic environments [14]. Typically, the environment is modelled as a Markov

CONTACT Konstantinos Blekas kblekas@cse.uoi.gr Department of Computer Science and Engineering, University of Ioannina, Ioannina, 45110 Greece

Supplemental data for this article can be accessed here. https://doi.org/10.1080/01691864.2020.1757507

Decision Process (MDP), where the agent receives a scalar reward signal that evaluates every transition. The objective is to maximize its long-term profit that is equivalent to maximizing the expected total discounted reward. Thus the learning process is designed on selecting actions with the optimum expected reward. Discovering optimal policy for agent is conducted with the notion of *value function* which associates every state with the expected discounted reward when starting from this state and all decisions are made following this policy. Q-learning algorithm [15] that belongs to the temporal difference family of methods constitutes one of the most popular mechanisms for building a RL agent among other [16]. An improved method has been recently presented in [17,18], called Double Q-learning, that prevents from overestimation of actions that may negatively affect the performance. An important advantage of double Q-learning is its beneficial effect on the stability of learning [17,18].

In the literature there are several works with RL frameworks in mini-robots applications. A mini-robot architecture together with reinforcement learning applications for obstacle avoidance and control are presented in [19]. In [20] a medical colon endoscope robot is presented that adjusts its locomotion through the use of tabular Q-learning RL scheme. Another application of the tabular Q-learning algorithm is found in [21] where a simple RL framework is used to an autonomous micro-robot. The algorithm is optimized in speed and memory consumption in order to be employed in mini-robots with low resources (memory, power, processor performance). A recent work is presented in [22] using a pair of AMiRo mini-robots with various sensor modalities that employs a RL-based distributed sensing framework based on latent space from multi-modal deep generative models. Also, in [23] a deep RL method was used to navigate micro/colloidal robots regarding obstacle avoidance and travel time minimization. An actor-critic multi-agent reinforcement learning approach is proposed in [24] based on deep Q-learning where the critic has access to the full state information, while the actor has local observations. Finally, deep multiagent reinforcement learning approaches have been developed recently with applications to swarm robotic systems [25–28], where many identical agents (mostly with limited actuation and sensors capabilities) interact with each other to achieve a common goal.

In this paper, we address the problem of autonomous navigation of a small group of multiple mini-robots (up to four), where we formulate it as a multiagent Markov Decision Process (MDP) based on a collaborative reinforcement learning framework [29–31]. The primary goal is to learn joint agent's policies to simultaneously navigate all mini-robots toward targets. The workspace is unknown for any robot and apart from static obstacles, the other mini-robots are considered as moving obstacles that must be avoided. A capable state space and an efficient reward function are introduced that aim at constructing agents' policies with optimal driving behavior that enable agents to reach their destinations safely and rapidly. Through the reward function a collaborative environment is constructed where mini-robots coordinate their strategies so as to execute their tasks jointly.

The output of the reinforcement learning framework is the desired direction of the velocity for each mini-robot. The required forces for the realization of the commanded velocities are provided by two vibration motors on each mini-robot. The dynamic model of the mini-robot, including the vibration motors dynamics, is integrated into the simulation environment. In order to compensate for unknown disturbances, and improve the motion resolution and the bandwidth of the mini-robot, a simple and low-cost PI speed controller for each vibration motor is implemented.

In this work we have studied both tabular Double and classical (single) Q-learning algorithms for training robots' policies in their discrete state space.

Several simulation examples considering a variety of scenarios with different degree of complexity have been made that show the effectiveness of the proposed framework. The simulation results from both Q-learning algorithms are presented and compared. According to the results, the method has the capability to successfully resolves any scenario and to achieve sub-optimum in length navigation paths. Furthermore, it manages to provide robust agents' policies with generalization capabilities over test environments. To our knowledge, this is the first time a collaborative multi-agent RL framework that employs the tabular double Q-learning algorithm is successfully implemented for multiple mini-robots for discovering their navigation policy.

Preliminary results assessing the performance of an abstract related scheme have been presented in [32] where some initial results were reported, and only two mini-robots were considered. With respect to this work the following issues summarize the novel material in this paper:

- An extended and improved multi-agent reinforcement learning framework is proposed, suited to the navigation of a group of several mini-robots in a collaborative environment. The tabular Double Q-learning algorithm is employed that offers improved robustness and performance in comparison with the classical Q-learning.

- A new and improved reward function is presented that allows coordination of mini-robots' strategies.
- A more complex simulation environment is developed including several mini-robots (four) and considering several alternative scenarios.
- Extensive simulations and experiments are conducted to evaluate and demonstrate the effectiveness of the proposed method with the addition of the Double Q-learning algorithm.

The remainder of this paper is organized as follows. Section 2 gives a brief description of application examples and scenarios studied in this paper, while Section 3 gives the reinforcement learning scheme developed for the navigation of a group of such mini-robots. The simulation scenarios and comparative results are presented in Section 4 and we conclude with a discussion and future directions in Section 5.

2. Application examples and scenarios

The proposed reinforcement learning framework can be applicable to any multi-agent system. Nevertheless, it is evaluated in a simulated environment comprising up to four mini-robots under a microscope. The mini-robots under consideration, presented in [12], are tetherless, fully autonomous and designed to perform micromanipulation and microassembly tasks, such as the cooperative fabrication of micro-systems or manipulation of biological specimens, in a micro scale environment. Several applications scenarios are implemented, where the goal was the navigation of the mini-robots to predefined positions, under the assumption of an unknown environment, in order to perform cell-manipulation or micro-assembly activities under the microscope.

Example simulation environments are shown in Figure 1(a and b). Since all robots share the same workspace they have to avoid collisions and navigate through limited space. The mini-robots targets are located in the central region of the board (Figure 1(b)) having enough space among them to perform manipulation tasks without collisions. An example of real-world scenario would be when different procedures are required to be performed in different positions and each robot can carry only one type of instrument. This can be done quickly with more robots rather than using a single mini-robot to perform multiple procedures.

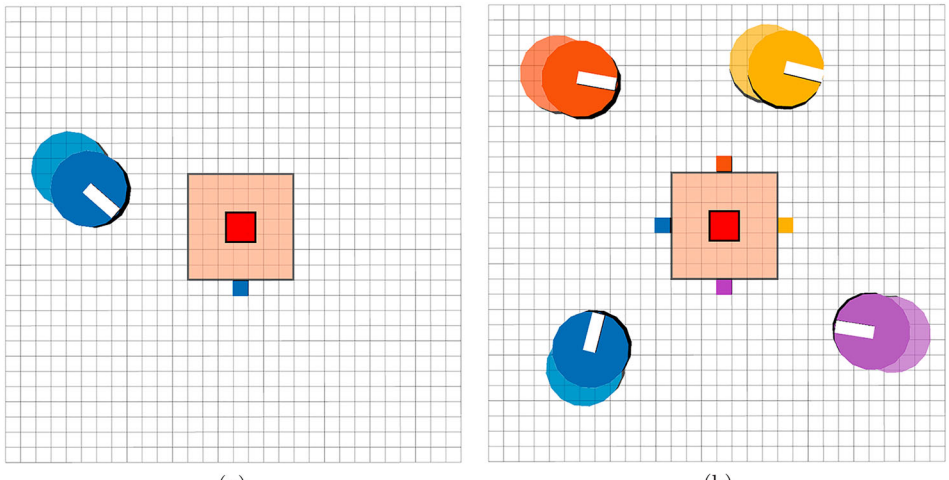
A functional prototype of the mini-robot is shown in Figure 2(a), see [33]. The mass of the mini-robot is equal to 0.1 Kg, with a body diameter and height of 0.05 m and 0.045 m respectively. The mini-robot is equipped with two vibration micro-motors with an eccentric mass. A PI speed controller is designed and implemented to improve the dynamic response of each micro-motor, and to compensate for the unknown disturbances, see [32]. The motion principle is based on the gravitational and centripetal forces exerted on each rotating eccentric mass. Above a critical value of actuation speed, actuation forces overcome frictional forces and motion is induced.

The dynamic model of the mini-robot is described by the following equations:

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{R}\sum_i {}^b\mathbf{f}_i$$

$${}^b\mathbf{I}\dot{\boldsymbol{\omega}}_p + {}^b\boldsymbol{\omega}_p \times {}^b\mathbf{I}^b\boldsymbol{\omega}_p = \sum_i \left({}^b\mathbf{r}_i \times {}^b\mathbf{f}_i\right) + \sum_j {}^b\mathbf{n}_j \quad (1)$$

where $i = \{A, B, C, D, E\}$, and $j = \{D, E\}$, see Figure 2(b). In (1), $\mathbf{M}$ denotes the mass of the mini-robot, $\mathbf{v} = [\dot{x}, \dot{y}, \dot{z}]^\top$ is the linear velocity of the center of mass of the mini-robot, and $\mathbf{R}$ is the rotation matrix from the



(a) (b)

Figure 1. Multi-robot navigation concept: (a) Single mini-robot environment; (b) Four mini-robots environment.

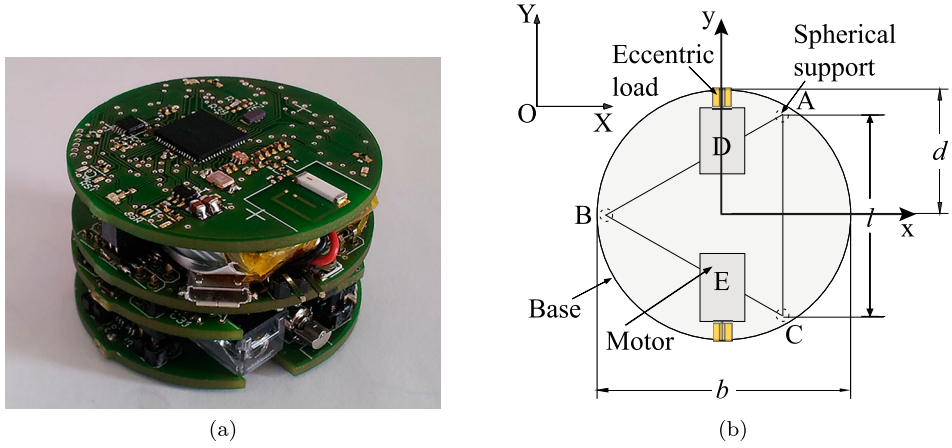


Figure 2. (a) Prototype; (b) Design concept (top view) (figures from [33]).

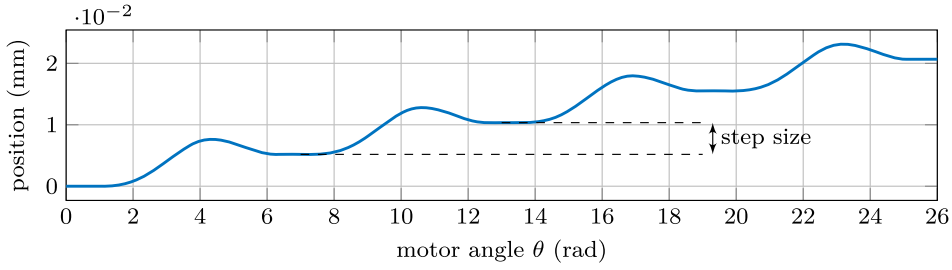


Figure 3. Simulation results for the net displacement toward the positive *X*-axis.

body frame, $\{b\}$, to the inertial frame. ${}^{b}\mathbf{f}_i$ is a vector that includes the actuation forces generated by the two vibration motors and the friction forces at the three contact points of the mini-robot, and ${}^{b}\mathbf{n}_j$ includes the moments exerted by the vibration motors. The moment of inertia of the mini-robot is denoted by $\mathbf{I}$, and $\boldsymbol{\omega}_p$ is the angular velocity of the mini-robot. Finally, ${}^{b}\mathbf{r}_i$ is the position vector of point i expressed in the body frame.

The synchronous rotation of the vibration motors of the platform, results in a net displacement toward the positive or the negative X-axis or a rotation about the vertical Z-axis, depending on the direction of rotation of each vibration motor, see Figure 3.

It has been shown analytically that the motion step the mini-robot exhibits over a cycle of operation can be made arbitrarily small depending on the actuation speed ω. For a more detailed presentation of the dynamics, design, and actuation principle of the mini-robot, see [12].

3. Reinforcement learning for mini-robots navigation

The Reinforcement Learning (RL) agent constitutes the basic building block of the proposed decision support system. The RL agent receives a state related to the position of the mini-robot platform and performs an action which corresponds to the direction of its velocity. Note that the desired magnitude of the platform's velocity is constant and not affected by the RL agent. In addition, the velocity and orientation of the platform are controlled by the low-level controller proposed in [32].

We have formulated the task as a *Markov decision process* (MDP) that constitutes an efficient mathematical framework for sequential decision making problems. It is represented as a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$, where:

- $\mathcal{S}$ denotes the set of agent's states. In our case we have considered the platform inertial coordinates. i.e. $s = (x, y)$ as the states of the agent. However, as it will be shown later the state vector will be enriched with other features in the case of treating multiple mini-robots.
- $\mathcal{A}$ is the action space. We have considered eight (8) discrete values: $\mathcal{A} = \{0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°\}$ that correspond to eight different directions of mini-robot velocity. It must be noted that this size of the action set seems to be appropriate in our application leading to satisfactory performance. However, any other set of actions can also be used.
- T denotes the Markovian state transition function that specifies the probability of visiting a new state s' from state s by taking action a, $P(s' \mid s, a) : S \times A \times S \rightarrow [0, 1]$.

- $R : \mathcal{S} \rightarrow \mathbb{R}$ is the reward function for a state-action pair, $R(s, a)$, describing the immediate reward of executing action a in state $\boldsymbol{s}$, and finally,
- γ is the discount factor across time ranging in $[0, 1]$ used for future rewards.

The decision mechanism of the agent is guided by the policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that is designed to perform a map from state space to actions. The state-action value function (Q-value function) of agent, $Q(s, a)$ describes the expected discounted reward received by starting from state s, executing the action a and following the policy π thereafter. According to the Bellman equations [14] the Q-function can be recursively written as:

$$\begin{aligned} Q^{\pi}(s, a) &= E_{\pi}\{R_t | s_t = s, a_t = a\} \\ &= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right\} \end{aligned} \tag{2}$$

A standard method for training an RL agent is by learning the Q-function via minimizing the Bellman's equation error, given by:

$$\begin{aligned} &\mathcal{E}_t(\theta) \\ &= \frac{1}{2} E_{s\sim T, a\sim\pi}\left\| R(\boldsymbol{s}_t, a_t) + \gamma \max_a Q(\boldsymbol{s}_{t+1}, a) - Q(\boldsymbol{s}_t, a_t)\right\|^2 \end{aligned} \tag{3}$$

The objective of RL problems is to estimate optimal policy π^* by choosing actions that yield the appropriate action-state value function, i.e.

$$\pi^*(\boldsymbol{s}) = \arg\max_{a\in\mathcal{A}} Q^{\pi}(\boldsymbol{s}, a) \tag{4}$$

The term 'optimal' is used to describe the shortest path from any starting position as well as the minimum required rotations of each mini-robot to achieve the commanded direction. The reason we also use the least number of direction changes is because of the limitation of the robot model we use, this robot requires a substantial amount of time to change the direction.

A common choice for training the agent is through Q-learning [15].

Q-learning is a standard off-policy method [14,34] where the agent learns about a policy different from the one it is executing. Policy-gradient methods [35], such as actor-critic methods [36], provide an alternative strategy where the critic is used to evaluate the current policy of the actor. For example, a learning method for autonomous object assembly tasks with swarm mini-robots (Kilobots) have been presented in [26], where the proposed reinforcement learning algorithm is based on the actor-critic relative entropy policy search (AC-REPS) algorithm, introduced in [37].

In Q-learning the following policy update rule takes place based on temporal-difference:

$$\begin{aligned} Q(s, a) \leftarrow & Q(s, a) \\ & + \eta\left[R(s, a) + \gamma \max_{a'} Q^{\pi}(s', a') - Q(s, a)\right] \end{aligned} \tag{5}$$

where η is the learning rate (during all simulation runs we set $\eta = 0.01$).

An alternative learning algorithm has been presented recentrly, called double Q-learning [18], that uses the double estimator to ease the overestimation. This is an off-policy value based reinforcement learning algorithm that employs a double estimator approach to determine the value of the next state. Double Q-learning tackles the problem of Q-value overestimation which arises from the fact that the Bellman equation computes the maximum of a noisy estimator. More specifically, tabular double Q-learning stores two functions, Q^A and Q^B. During training first a single Q function is randomly selected (Q^A or Q^B) and then it is updated with a value from the other function for the next state. The next updated equations are taken place:

$$\begin{aligned} Q^A(s, a) \leftarrow & Q^A(s, a) \\ & + \eta\left[R(s, a) + \gamma \max_{a'} Q^B(s', a') - Q^A(s, a)\right] \end{aligned} \tag{6}$$

$$\begin{aligned} Q^B(s, a) \leftarrow & Q^B(s, a) \\ & + \eta\left[R(s, a) + \gamma \max_{a'} Q^A(s', a') - Q^B(s, a)\right] \end{aligned} \tag{7}$$

As indicated in [18], both Q functions learn from separate sets of experiences, however they are both used for the action selection mechanism: action is estimated according to the ϵ-greedy framework by calculating the average of the two Q values.

The reward signal, $R(s, a)$, for a single mini-robot is simply defined as following:

$$R(s, a) = \begin{cases} +L, & \text{if it reaches goal} \\ -L, & \text{out of border} \\ -1, & \text{otherwise} \end{cases} \tag{8}$$

where L is a positive constant term; it was set to $L = 100$.

Each episode typically starts at a random position on the edge of the board. During learning the agent is trying to explore the environment by collecting samples and gradually uses them for exploitation rather than exploration. An important issue in reinforcement learning is how to manage the trade-off between exploration and exploitation since it may have significant impact to the quality of learned policy. A common practice is to

employ the ϵ-greedy exploration scheme, where at each time step t an action is selected greedily, based on the estimated action-value function with probability $1 - \epsilon_t$, while a random action is chosen with probability ϵ_t, ($\epsilon_t \in [0, 1]$). During the simulation runs the parameter ϵ_0 is initially set to a large value (e.g. $\epsilon_0 = 0.9$), and it decays exponentially over time (decay rate 0.999).

Finally, it must be noted that an *episode* is defined as a sequence of transitions that terminates either when the agent reaches the goal state, or when the mini-robot platform collides with the obstacle which is the restricted zone in the middle of the board. We consider this obstacle to be an object for the mini-robot to do an operation on, e.g. biological specimen. The other terminal state of an *episode* is when the agent goes out of the workspace (board). Note that in this application we consider that there is no additional obstacle so the agent is free to explore through the entire board without any obstruction.

3.1. MDP formulation for multiple mini-robots

The primary aim of this work is to develop a reinforcement learning framework for modeling multiple mini-robots which are jointly interacting with the environment in a collaborative manner. Each robot must learn a suboptimal policy so as to reach its goal position under the presence of other robots. The impact of this is that all goal positions of the other mini-robots correspond to additional *static* obstacles for any robot. Furthermore, while navigation other agents must be considered as *dynamic* obstacles which appeared without any predefined knowledge of their behaviour and must be avoided.

We are focused on constructing proximity collision avoidance policies for all mini-robots considering centralized control so as to perceive the position of nearby agents and calculate the rewards. The multiple robots navigation system utilizes the basics from single robot environment with added features derived from the co-occurrence of all agents. The workspace is a rectangular flat surface and is divided into grids. All agents don't have any prior knowledge of the environment and their starting positions are different and random. During the simulation runs we assume various scenarios with variable robots' targets and starting position areas on the workspace. An episode ends when, either all mini-robots reach their goals, or at least one agent fails, i.e. it goes out of border, or collides with an (static or dynamic) obstacle.

Each agent has its own goal situated at a different position and occupied a single grid cell. An example can be seen in Figure 1(b). As mentioned before, all robots share the same workspace, therefore they have to avoid collisions and navigate through limited space. The robot targets are located in the central region of the board (Figure 1(b)) having enough space among them to avoid collisions.

Except for its inertial coordinates (x_i, y_i), the state space of every mini-robot i also includes two other quantities:

- The *minimum distance* d_i with all other robots. As shown in Figure 4, this distance has been discretized into three (3) constant values: 'near' ($5 < d_i \leq 6$), 'medium' ($6 < d_i \leq 10$), 'far' ($d_i > 10$).
- The *sign of the distance gradient*, g_i, that indicates whether the distance between the nearest mini-robot is increased or decreased at each time step, t:

$$g_i = \text{sign}(d_i^{(t)} - d_i^{(t-1)}) \tag{9}$$

Thus the state variable is represented as a vector of four quantities:

$$s_i = (x_i, y_i, d_i, g_i). \tag{10}$$

This allows the agent to realize the presence of moving obstacles (other robots) in its neighborhood so as to manage the creation of a policy for avoiding them. It must be noted that the discretization of distance can be adjusted to have appropriate size and distance for the workspace and complexity of the problem.

3.1.1. The proposed reward function

The reward function evaluates the selected action based on the current mini-robot's state. In our study we have designed an efficient reward function that aims at eliminating the collisions and minimizing the travelling path of all mini-robots and the time taken to reach their destination and discovering their targets.

The proposed reward function is formulated as:

$$R(s_i, a_i) = \begin{cases} L, & \text{reaches goal} \\ -L, & \text{out of workspace, collision} \\ -b_i \left[k * \max_j \left(1 - \frac{dist_{i,j}}{D} \right) \right] - c, & \text{otherwise} \end{cases} \tag{11}$$

where

- $dist_{ij}$ is discrete distance between two robots.
- D is maximum discrete distance in workspace.
- L, k, c are positive constants: The first term was set to $L = 1000$). As we will see later in our simulations we have made an experimental study where we have measured the impact of the last two parameters, k, c, and we selected the optimum values of them.

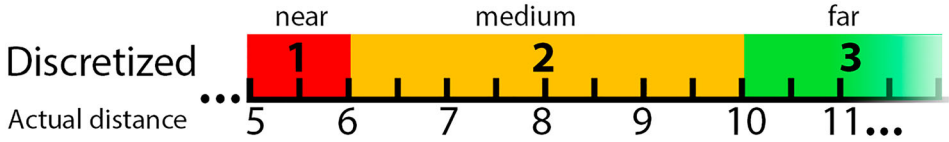


Figure 4. Distance discretization.

- b_i is distance gradient penalty coefficient, conditionally dependent on the sign of g_i.

$$b_i = \begin{cases} 1.5 & \text{if } g_i < 0 \\ 1.0 & \text{if } g_i \geq 0 \end{cases} \tag{12}$$

An explanation of the above formulation is the following:

- Likewise the single mini-robot case, the received reward is constantly positively large ($+L$) when the mini-robot reaches its goal, and negatively large ($-L$) when it finds an obstacle or is located outside the grid border.
- The bias term c provides the default reward when the agent has not reached the target yet without having any other mini-robot in its visual field and indicates the maximum allowed value.
- The distance gradient penalty coefficient, b_i, penalizes more ($b_i \geq 1$) in the situation where its nearest mini-robot is approaching. It can be considered as a weight for the direction of a mini-robot.
- Finally, the parameter k is referred to the nearest neighbor distance and provides a weight of this distance to the reward function.

4. Simulation results

We have studied the performance of the proposed method by conducting several simulated scenarios. The simulation environment has been implemented using the ROS (Robot Operating System) framework. It includes the kinematic and dynamic model of the mini-robotic platforms, information about the proximity between the robots and white noise added to each vibration motor rotational speed that is equivalent to ±5% of the nominal rotational speed. In all simulation runs the integration time step for the mini-robotic platforms was set to $dt = 10^{-5}$, while the step of RL agent is equal to 2×10^5 integration time steps, equivalent to 2 seconds. The workspace is 30×30 cm, with a rectangular restricted zone of size 7×7 cm located at its center. It is assumed that every target position of each mini-robot occupies a single cell next to the border of the restricted zone at the central region of the board.

During the execution of the multiagents reinforcement learning framework we have compared both Q-learning algorithms, the single (*Single Q*) and the double Q-learning (*Double Q*), for training the agents. In all experiments we have used a discount rate of $\gamma = 0.99$ and a learning rate parameter of $\eta = 0.01$. Furthermore, we have considered the ε-greedy exploration scheme with an initial probability of $\varepsilon = 0.9$ together with a linear reduction scheme using a coefficient of 0.999 at every 2500 and 500 episodes, respectively for both algorithms. In total 25×10^6 (Double Q method) and 5×10^6 (Single Q method) episodes were required for training the agents that correspond to an execution time about 40 and 15 min, respectively[1]. The initial 80% of them were used for the exploration phase in both methods.

We evaluated all policies generated by both methods using the following two metrics:

- the percentage of the successful episodes, and
- the required average distance (path length) to reach the mini-robots' targets when starting from a random position of the workspace border.

During all scenarios we calculated the mean values of each performance metric after executing 20 independent simulation runs.

4.1. Scenarios description

In figures 5, 8 and 9 we illustrate the scenarios used for one and four mini-robots along with their target positions. Each mini-robot has to be trained to find safely the destination to its respective target marked by the same color as the robot. In the case of a single mini-robot we have considered the scenario where the mini-robot starts randomly from a zone of width equal to one cell on the perimeter of the board (see Figure 5(a)). The goal is located at the cell coordinate (15, 11), under the restricted zone. In the case of four mini-robots we have applied six scenarios with different starting area for each mini-robot, as given in Table 1 and illustrated in Figures 8 and 9. The goal for each robot is presented as $G1$, $G2$, $G3$ and $G4$, respectively. In the last scenario *(f)* all mini-robots start at random positions around the perimeter of the

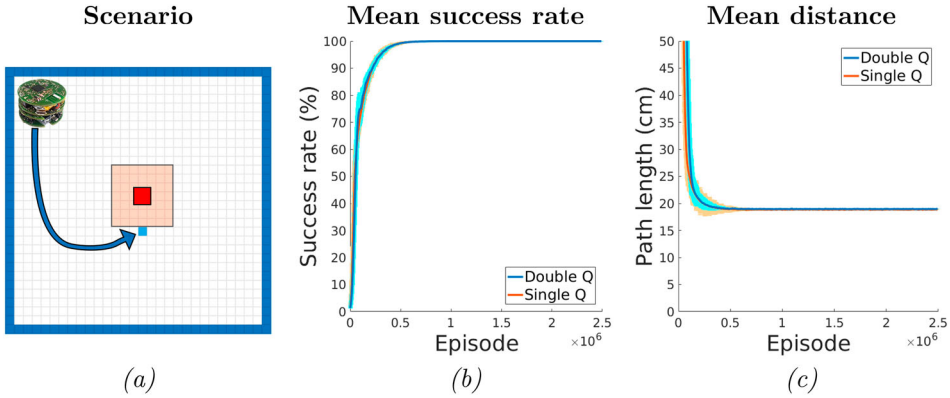


Figure 5. The simulated scenario used and the obtained comparative learning curves of the single mini-robot training. The curves represent the mean success rate and the mean path length needed to reach the target of double and single Q-learning algorithms, together with the standard deviation.

Table 1. Description of the scenarios with 4 min-robots used in our simulation runs. *Si* denotes the (x, y) coordinates of the *i*th mini-robot's starting position.

	Mini-robots starting position (cm)			
Scenario	*S1*	*S2*	*S3*	*S4*
(a)	$(1 \pm 1, 15 \pm 2.5)$	$(15 \pm 2.5, 29 \pm 1)$	$(29 \pm 1, 15 \pm 2.5)$	$(15 \pm 2.5, 1 \pm 1)$
(b)	$(15 \pm 2.5, 1 \pm 1)$	$(1 \pm 1, 15 \pm 2.5)$	$(15 \pm 2.5, 29 \pm 1)$	$(29 \pm 1, 15 \pm 2.5)$
(c)	$(29 \pm 1, 15 \pm 2.5)$	$(15 \pm 2.5, 1 \pm 1)$	$(1 \pm 1, 15 \pm 2.5)$	$(15 \pm 2.5, 29 \pm 1)$
(d)	$(3 \pm 1, 1 \pm 1)$	$(11 \pm 1, 1 \pm 1)$	$(19 \pm 1, 1 \pm 1)$	$(27 \pm 1, 1 \pm 1)$
(e)	$(3 \pm 1, 1 \pm 1)$	$(11 \pm 1, 1 \pm 1)$	$(3 \pm 1, 29 \pm 1)$	$(11 \pm 1, 29 \pm 1)$
(f)	*random*			

workspace, assuming that there is a distance of at least 10*cm* between each mini-robot in the beginning.

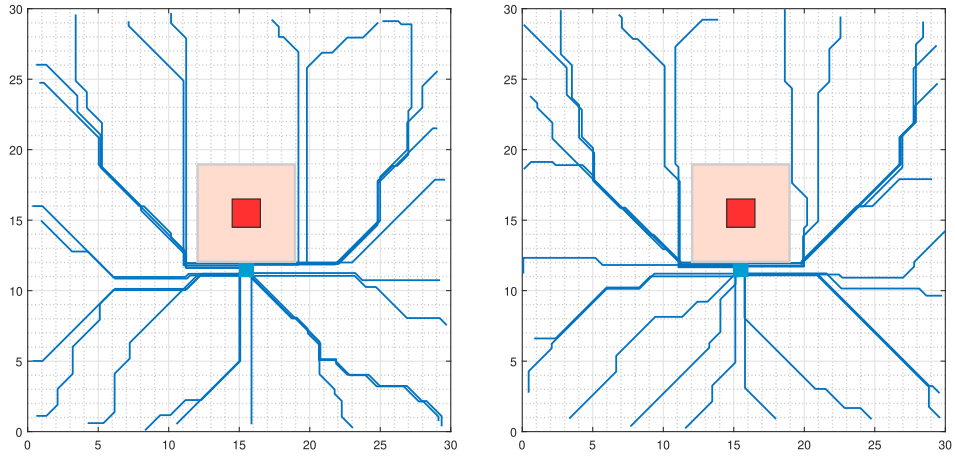
4.2. Simulation with a single mini-robot

At first we examined the performance of both algorithms for training the agent, Double and Single Q-learning, to the simulated environment in the case of a single mini-robot, see Figure 5(a). In Figure 5(b and c) we present the (mean) learning curves received during training the single mini-robot agent in 20 independent simulation runs using both Q-learning schemes. In particular we show the mean and the standard deviation of the success rate (percentage of reaching the target) and the required path length (in cm). As shown, the performance of both methods is almost identical converging effectively (100% success rate) and quickly after a short period of exploration. It is interesting to note their ability to reach always the same solution with a very low variability.

In addition, in Figure 6 we show several robot navigation paths obtained by both Q-learning algorithms considering various random positions around the borders of the workspace. Based on the results it is worth mentioning that the proposed frameworks have the ability to estimate sub-optimal paths by any random starting position. We have found an average path length of around 19 cm (50 agent's steps) for both Double and Single Q-learning algorithms, while executing the task of the mini-robot navigation 1000 times after finishing the training procedure.

4.3. Simulation with multiple mini-robots

The primary aim of this work is to apply multiagent systems to the task of navigating multiple mini-robots in unknown environments. We start our study by initially evaluating the impact of the reward function's parameters k and c (Equation (11)) to the performance of the proposed method. As mentioned before, the first coefficient, k, is used to penalize the distance with the nearest mini-robot, while parameter c plays the role of a threshold value for the reward function and denotes the maximum allowed reward value a mini-robot can receive. In our study we have examined four (4) different values of the k parameter, $k = \{0.5, 1.0, 5.0, 10.0\}$ and seven (7) different values of the c parameter, $c = \{0.25, 0.5, 0.75, 1.0, 2.0, 3.0, 5.0\}$, i.e. in total twenty (28) pairs of values were evaluated. The obtained results are presented in Figure 7 in the case of the Single Q-learning scheme. Two evaluation metrics were used: mean success rate and mean estimated path length estimated in the scenario *(f)* that considers random starting positions of mini-robots and constitutes the most difficult



(a) Double Q-learning (b) Single Q-learning

Figure 6. Exemplar mini-robot trajectories obtained after finishing the learning procedure with both Q-learning algorithms.

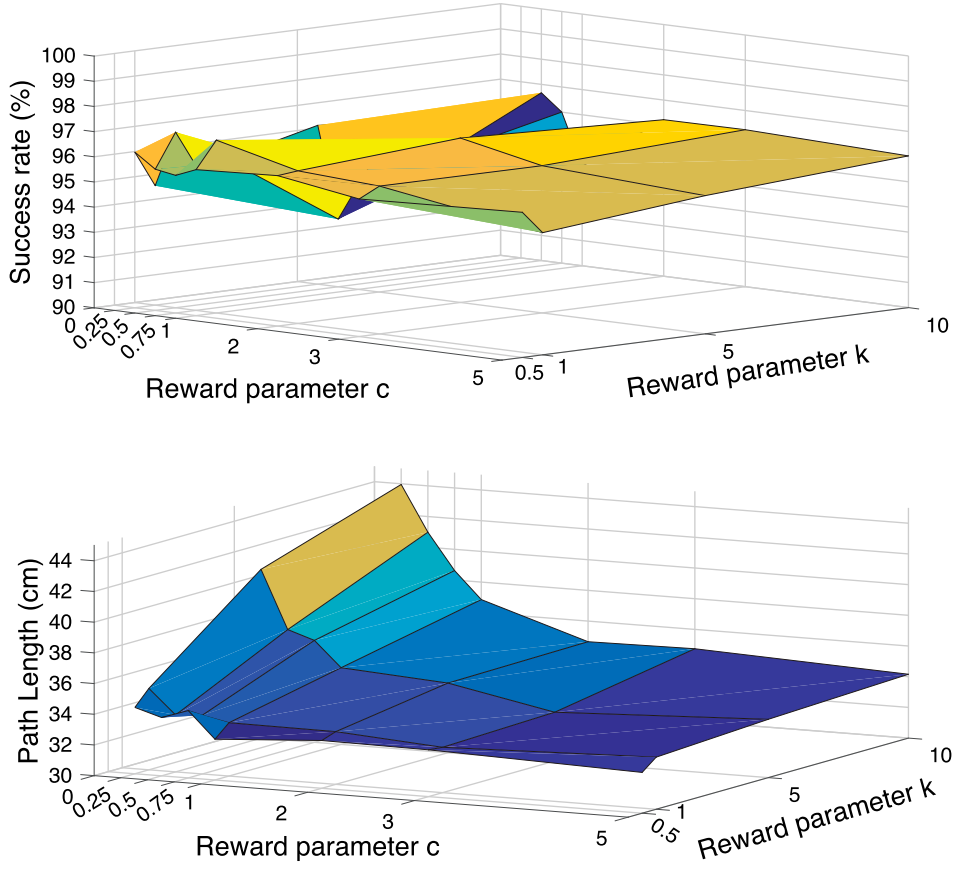


Figure 7. Evaluation of the impact of the reward function parameters k and c (Equations (11)) on the scenario *(f)*. The diagrams show the *mean percentage of the successful episodes* and the *mean path length* taken from 1000 episodes of 20 learned policies obtained after executing 20 independent simulations. All simulation runs were made using the Single Q-learning scheme in the last scenario (f) that constitutes the most difficult case.

case. They are calculated by executing 20 independent simulated runs and then 1000 episodes at every multi-agent policy discovered. According to both graphs it is interesting to observe that the method exhibits almost constant behavior in terms of percentage of successful episodes (success rate) and local variability in terms of

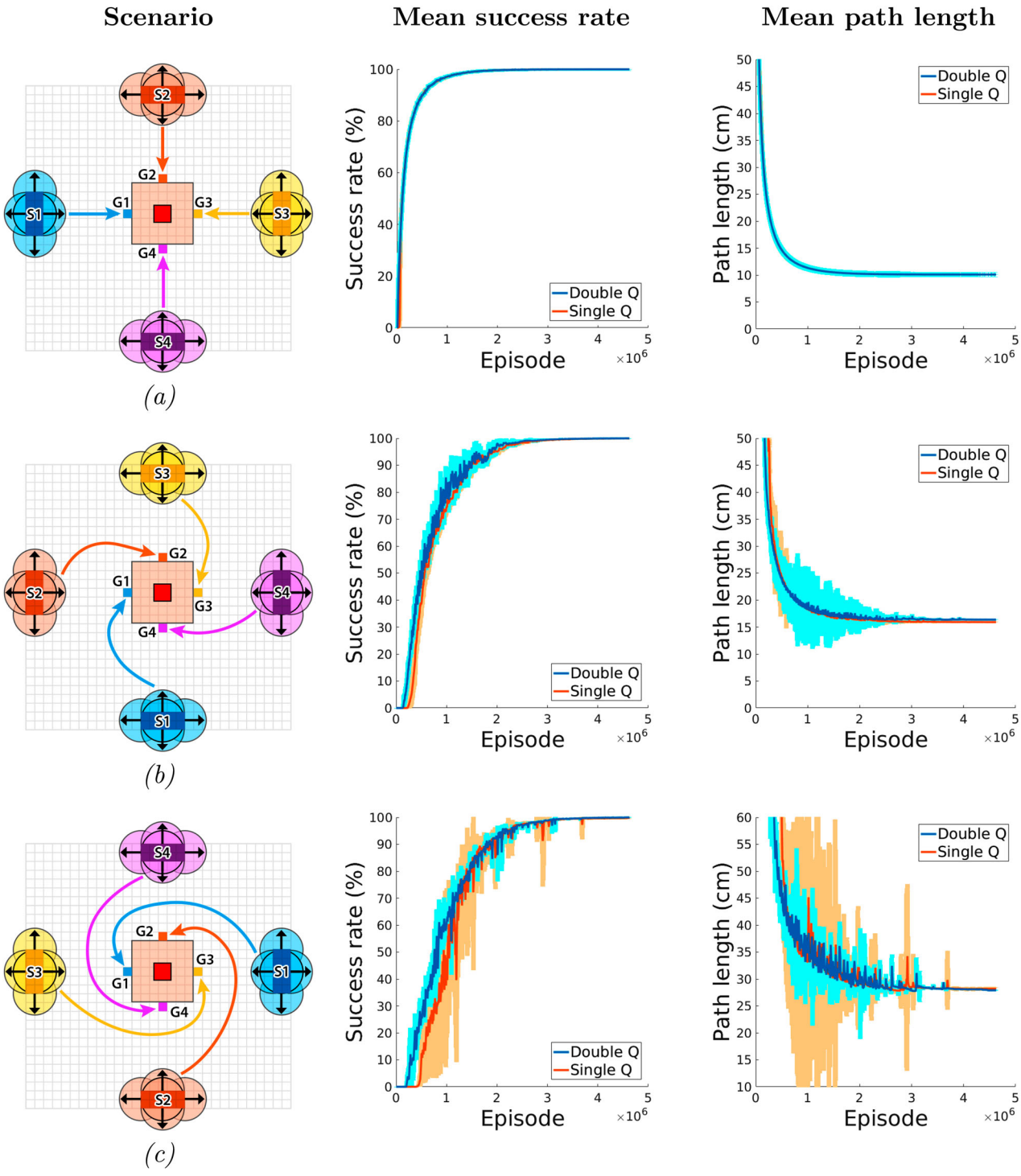


Figure 8. Comparative learning curves from the application of the proposed multiagent RL approach using the Double and Single Q-learning algorithm, respectively, to three different simulated scenarios *(a), (b), (c)*. The curves represent the mean values and the standard deviations of the success rate and the mean path length required from four mini-robots to reach the targets as obtained by 20 independent simulation runs. The target positions are the same in all cases, while the starting positions of four mini-robots differ at every scenario.

path length. We obtained better results with $k = 1.0$ and $c = 0.5$. Finally, it must be noted that we acquire similar results in the case of Double Q-learning scheme and therefore we have adopted the above values for both reward function parameters throughout the remaining simulation runs.

In our study we have considered six different simulated scenarios with variable difficulty presented in

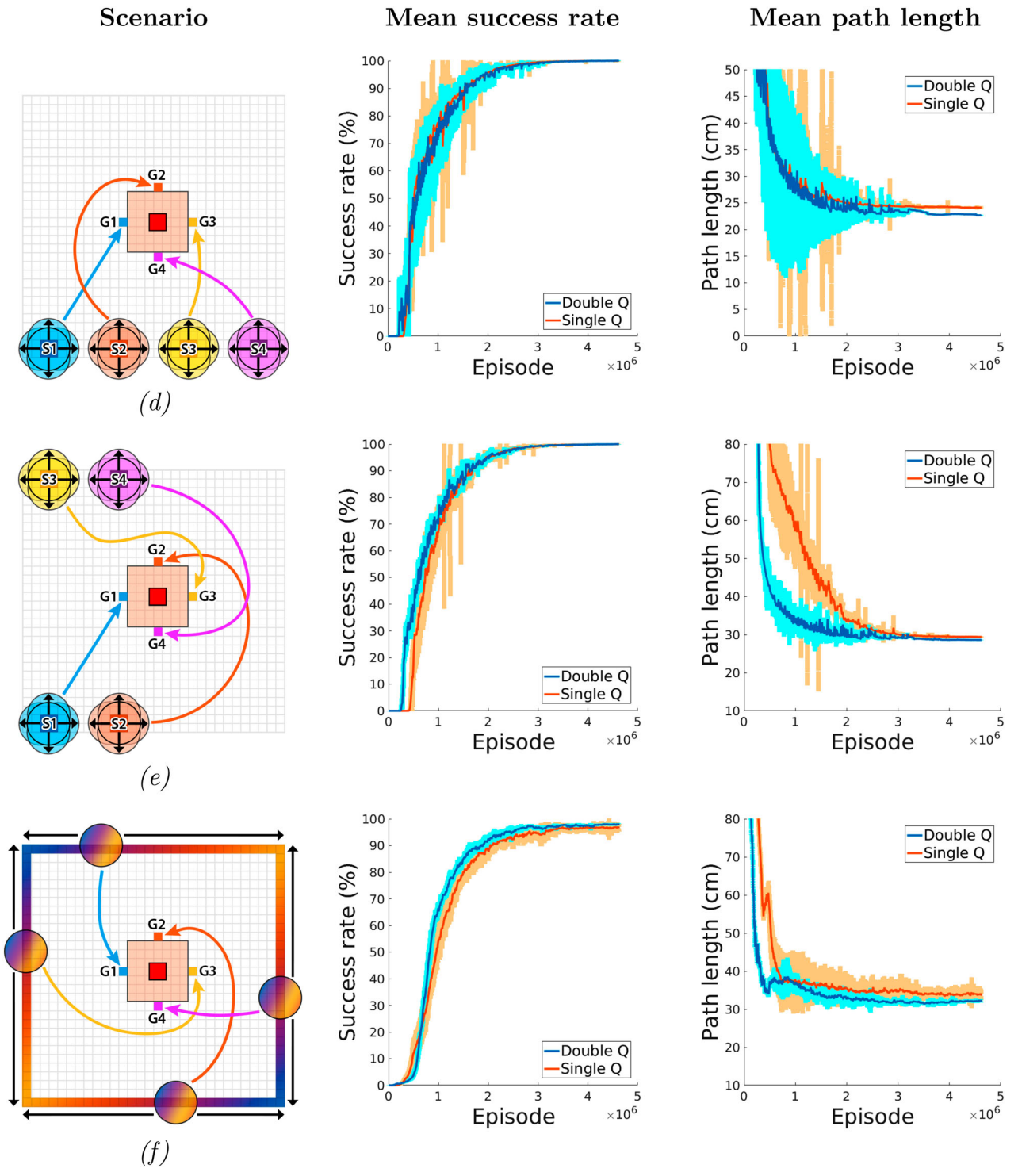


Figure 9. Comparative learning curves from the application of the proposed multiagent RL approach using the Double and Single Q-learning algorithm, respectively, to three different simulated scenarios *(d), (e), (f)*. The curves represent the mean values and the standard deviations of the success rate and the mean path length required from four mini-robots to reach the targets as obtained by 20 independent simulation runs. The target positions are the same in all cases, while the starting positions of four mini-robots differ at every scenario.

Figure 8 and 9. In the first scenario *(a)* the target of each mini-robot is just opposite to its starting position and thus the probability (a priori) of collision is low. The level of difficulty is progressively increased in the next five scenarios (scenario *b* to *f*) and the environment becomes more complicated in terms of the starting position of each mini-robot and subsequently the determination of the navigation paths to targets which might cross with other

mini-robots. Therefore, each mini-robot's policy must be learned by interacting with the environment so as to reach its target, as well as to avoid the collision with the others. Furthermore, all agents must act collaboratively in order to resolve the task together.

Figures 8 and 9 present the learning curves of the proposed multiagent RL methodology to every evaluation case, as calculated by executing 20 independent simulated runs using the Double and Single Q-learning algorithms. In particular, we show the obtained mean value and the standard deviation of the success rate and the path length (in cm) of all mini-robots. It must be noted that once a robot reaches its target, it remains there until all mini-robots terminate their motion (either successfully, or not). An episode is considered as successful only when all four mini-robots manage to reach their targets without any collision. Regarding the received learning curves we can observe that the performance of the Double Q-learning algorithm is slightly better, since in some cases it manages to create policies with increased

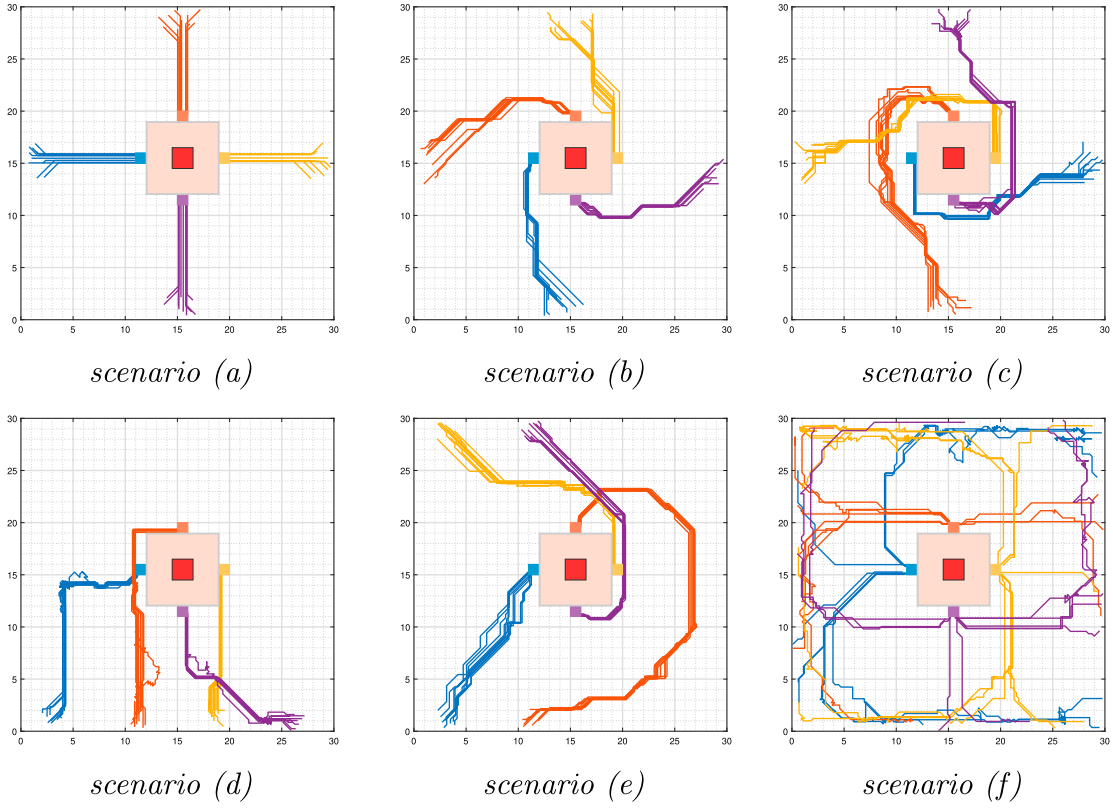


Figure 10. Exemplar trajectories for four mini-robots as obtained from the learned policies of the proposed multiagent RL approach using the Double Q-learning algorithm.

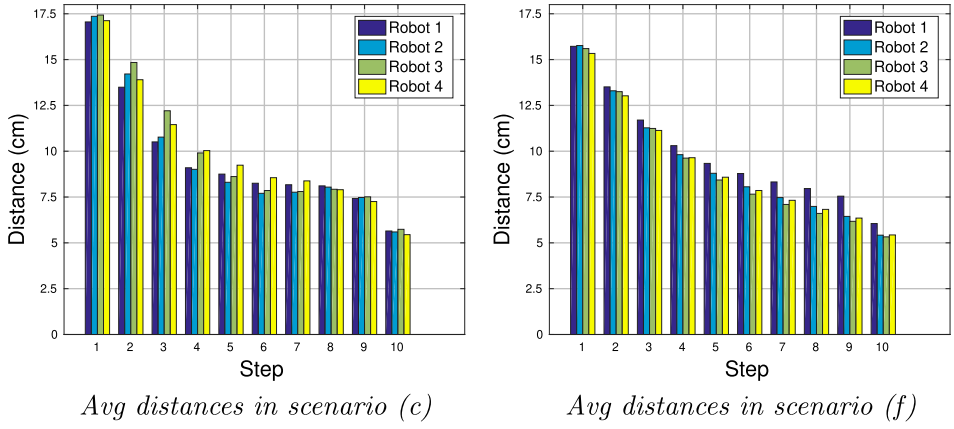


Figure 11. Average distances between each robot and their nearest neighbour during their trajectories (divided into 10 bins). The bars are obtained after executing 20 independent simulated runs in scenarios (c) and (f).

Table 2. The comparative test performance of the proposed multiagent RL framework using the Double and the Single Q-learning algorithm, in terms of two evaluation criteria in every scenario. Statistics are calculated after executing 20 simulations of 1000 episodes per case.

	Double Q		Single Q	
Scenario	Success rate (%)	Path length (cm)	Success rate (%)	Path length (cm)
(a)	$100\% \pm 0$	10.09 ± 0.01	$100\% \pm 0$	10.09 ± 0.01
(b)	$100\% \pm 0$	15.59 ± 0.14	$99.97\% \pm 0.03$	15.94 ± 0.08
(c)	$99.93\% \pm 0.03$	27.94 ± 0.46	$99.87\% \pm 0.11$	28.17 ± 0.74
(d)	$100\% \pm 0$	22.24 ± 1.87	$99.99\% \pm 0.05$	24.11 ± 0.81
(e)	$100\% \pm 0$	28.63 ± 0.64	$99.97\% \pm 0.09$	29.41 ± 0.96
(f)	$97.77\% \pm 0.69$	32.15 ± 0.37	$96.97\% \pm 1.09$	33.71 ± 0.79

success rate and lower path length. Another interesting observation is that the Double Q-learning seems to generate more compact solutions and near optimal policies compared to the Single Q method, since it has lower variability in both evaluation metrics during the learning process. Note that only in two scenarios (Figures 8(c) and 9(f)) the percentage of successful episodes was slightly lower than 100%.

In Figure 10 we illustrate several successful navigation paths for all mini-robots in all simulated scenarios based on the learned robot policies using the Double Q-learning algorithm. Seemingly, the proposed multiagent framework has the ability to efficiently learn joint agents' policies and to support mini-robots to reconcile conflicting decisions. The constructed paths of every mini-robot are designed to be sub-optimal and safe without the presence of any collision. In addition, in Figure 11 we evaluate further the produced robot paths of the learned policies in scenarios (c) and (f). In particular, we present the distance between each robot and its nearest one during executing their trajectories. Note that, since the navigation paths are of variable size, we have divided them into 10 bins, and we calculated the average distances of the successful simulation runs (i.e. no collision is occurred). The bars in Figure 11 are depicted after 50 independent simulation runs in scenarios (c) and (f). It is interesting to observe that during executing their trajectory all mini-robots are in save distance among them, since their minimum distance in both scenarios is always larger than $5cm$ which is the body diameter of each mini-robot. This illustrates both the quality of the constructed policies and the additional benefits of coordination among agents the proposed multiagent RL approach with Double Q-learning offers.

Finally, in Table 2 we present a summary of the comparative results we obtained employing the Double and the Single Q-learning algorithm to the proposed multiagent RL approach. 20 learned policies were used in a series of 1000 episodes per case, where we calculated the mean value and the standard deviation of the success rate and the path length from random starting positions. As in the learning process, the Double Q-learning seems to perform better in test environments with improved generalization capabilities. Again, only in two scenarios, (c) and (f), the success rate is slightly lower than perfect, while the Double Q-learning achieves less variability in path length which can successfully improve system's reliability.

5. Conclusions and discussion

In this study we have established an enhanced multiagent reinforcement learning framework for addressing the problem of the autonomous navigation of multiple mini-robots. The classical and the double Q-learning algorithms are employed with promising results, where the double Q-learning algorithm shows a slightly better performance. The key aspect of the proposed scheme lies on the efficient design of input state space of mini-robots that allows the creation of a collaborative environment among agents. Also, an advanced reward function was created that manages to control multiple mini-robots and navigate them safely and effectively to their target position by minimizing their travelling time.

We are in the middle of the design and implementation process of such mini-robots, and we expect to conduct experiments with real mini-robots in the next few months. Concerning the practical implementation we use a centralized approach, where each mini-robot would be equipped with a micro-processor that is responsible for the low- level desired function of the robot (power management, sensors and actuators functionality) and the low-level speed control. In addition, each mini-robot has the capability of wireless communication with a central computer where the high-level RL agents are implemented.

We aim at further pursuing and developing the proposed method in three main directions:

- Study the possibility to handle continuous state and action spaces and employ value-function approximation schemes.
- Extend the presented framework to deep reinforcement learning schemes [38,39].

- Validate the proposed method in more complex environments with greater degree of uncertainty, as well as in real-world scenarios using real mini-robots that are currently under construction.

Note

1. All experiments were conducted on an Intel Core i5-4950 PC with 16GB RAM under the C++ environment.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Piyabhum Chaysri received the B.Sc. degree in Computer Science in 2016 and received the M.Sc. degree in Technologies and Applications in 2018, both from the University of Ioannina, Ioannina, Greece. He currently continues his Ph.D. research in Reinforcement Learning Agents in Robotic Applications at the same university. His research interests include machine learning, intelligent agents, reinforcement learning, navigation and control of robotic platforms as well as designing and construction of robotic mechanisms.

Konstantinos Blekas received the Diploma degree in Electrical Engineering in 1993 and the Ph.D. degree in Electrical and Computer Engineering in 1997, both from the National Technical University of Athens. He is currently associate professor at the Department of Computer Science & Engineering, University of Ioannina, Greece. He has co-authored more than 90 refereed journal and conference articles. His research interests include machine learning, pattern recognition, intelligent agents and reinforcement learning with applications to robotics and autonomous systems, computer vision, and medicine.

Kostas Vlachos received the B.Sc. degree in electrical engineering from the Technical University of Dresden, Dresden, Germany, in 1993. He received from the National Technical University of Athens, Athens, Greece, the M.S. (2000) and Ph.D. (2004) degrees in the area of Automatic Control and Robotics respectively. From 1996 to 1998, he worked as a Software Analyst in INTRACOM S.A. From 2007 to 2013, he was a visiting Lecturer at the Mechanical Engineering Department, University of Thessaly, where he taught courses in the areas of Control Systems, and Robotics. Currently, he is an Assistant Professor with the Department of Computer Science and Engineering, University of Ioannina. Research interests: haptic mechanisms, medical simulation, microrobotics, navigation and control of robotic mechanisms.

ORCID

Kostas Vlachos http://orcid.org/0000-0002-9716-7661

References

[1] Kortschack A, Shirinov A, Truper T, et al. Development of mobile versatile nanohandling microrobots: design, driving principles, haptic control. Robotica. 2005;23: 419–434.

[2] Schneider A, Paskarbeit J, Schaeffersmann M, et al. Hector, a new hexapod robot platform with increased mobility – control approach, design and communication. *Advances in Autonomous Mini Robots – Proceedings of the 6th AMiRE Symposium*; 2012. p. 249–264.

[3] Theisgen L, Fuente M, Radermacher K. Modular design of versatile surgical mini-robots. Curr Dir Biomed Eng. 2018;4:411–414.

[4] Breguet JM, Clavel R. Stick and slip actuators: design, control, performances and applications. Int. Symposium on Micro-mechatronics and Human Science (MHS); 1998. p. 89–95.

[5] Schmoeckel F, Fatikow S. Smart flexible microrobots for scanning electron microscope (SEM) applications. J Intell Mater Syst Struct. 2000;11: 191–198.

[6] Buchi R, Zesch W, Codourey A. Inertial drives for micro- and nanorobots: analytical study. SPIE Photonics East '95: Proc. Microrobotics and Micromachanical Systems Symposium; 1995.

[7] Sylvain M, et al. Three-legged wireless miniature robots for mass-scale operations at the sub-atomic scale. IEEE International Conference on Robotics & Automation; 2001. p. 3423–3428.

[8] Brufau J, Puig-Vidal M, Lopez-Sanchez J, et al. MICRON: small autonomous robot for cell manipulation applications. Proc. of the IEEE International Conference on Robotics & Automation; 2005.

[9] Vartholomeos P, Loizou S, Thiel M, et al. Control of the multi agent micro-robotic platform micron. IEEE International Conference on Control Applications; 2006. p. 1414–1419.

[10] Rubenstein M, Ahler C, Nagpal R. Kilobot: a low cost scalable robot system for collective behaviors. 2012 IEEE International Conference on Robotics and Automation; 2012. p. 3293–3298.

[11] Herbrechtsmeier S, Korthals T, Schopping T, et al. Amiro: a modular customizable open-source mini robot platform. 2016 20th International Conference on System Theory, Control and Computing (ICSTCC); 2016. p. 687–692.

[12] Vartholomeos P, Papadopoulos E. Dynamics, design and simulation of a novel microrobotic platform employing vibration microactuators. J Dyn Syst Meas Contr. 2006;128:122–133.

[13] Vlachos K, Papadimitriou D, Papadopoulos E. Vibration-driven microrobot positioning methodologies for non-holonomic constraint compensation. Engineering. 2015; 1:066–072.

[14] Sutton RS, Barto AG. Reinforcement learning: an introduction. Cambridge: MIT Press; 2014.

[15] Watkins CJ, Dayan P. Q-learning. Mach Learn. 1992;8: 279–292.

[16] Szepesvari C. Algorithms for reinforcement learning. Morgan and Claypool Publishers; 2009.

[17] Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-Learning. Proc. of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI); 2016. p. 2094–2100.

[18] Hasselt HV. Double q-learning. Advances in Neural Information Processing Systems (NIPS) 23; 2010. p. 2613–2621.

[19] Pedre S, Cristóforis PD, Caccavelli J, et al. A mobile mini-robot architecture for research, education and pop-

ularization of science. J Appl Comput Sci Methods. 2010;2:41–59.
[20] Trovato G, Shikanai M, Ukawa G, et al. Development of a colon endoscope robot that adjusts its locomotion through the use of reinforcement learning. Int J Comput Assist Radiol Surg. 2010;5:317–325.
[21] Asadpour M, Siegwart R. Compact q-learning optimized for micro-robots with processing and memory constraints. Rob Auton Syst. 2004;48:49–61.
[22] Korthals T, Leitner J, Rückert U. Coordinated heterogeneous distributed perception based on latent space representation. CoRR abs/1809.04558. 2018.
[23] Yang Y, Bevan M, Li B. Efficient navigation of colloidal robots in an unknown environment via deep reinforcement learning. Adv Intell Syst. 2019;2.
[24] Lowe R, WU Y, Tamar A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems (NIPS) 30*; 2017. p. 6382–6393.
[25] Gebhardt G, Daun K, Schnaubelt M, et al. Learning robust policies for object manipulation with robot swarms. IEEE International Conference on Robotics and Automation (ICRA); 2018. p. 7688–7695.
[26] Huttenrauch M, Sosic A, Neumann G. Deep reinforcement learning for swarm systems. J Mach Learn Res. 2019;20:1–31.
[27] Iima H, Kuroe Y. Swarm reinforcement learning method for a multi-robot formation problem. IEEE International Conference on Systems, Man, and Cybernetics; 2013. p. 2298–2303.
[28] Sosic A, KhudaBukhsh W, Zoubir A, et al. Inverse reinforcement learning in swarm systems. International Conference on Autonomous Agents and Multiagent Systems; 2017. p. 1413–1421.
[29] Guestrin C, Lagoudakis M, Parr R. Coordinated reinforcement learning. International Conference on Machine Learning (ICML '02); 2002. p. 227–234.
[30] Kochenderfer MJ. Decision making under uncertainty: theory and application. MIT press; 2015.
[31] Kok JR, Vlassis N. Collaborative multiagent reinforcement learning by payoff propagation. J Mach Learn Res. 2006;7:1789–1828.
[32] Chaysri P, Blekas K, Vlachos K. Navigation of inertial forces driven mini-robots using reinforcement learning. Proc. of the IEEE Intern. Conf. on Information, Intelligence, Systems and Applications (IISA); 2019.
[33] Vartholomeos P, Vlachos K, Papadopoulos E. Analysis and motion control of a centrifugal-force microrobotic platform. IEEE Trans Autom Sci Eng. 2013;10:545–553.
[34] Degris T, White M, Sutton R. Off-policy actor-critic. Proc. of the 29th International Conference on Machine Learning (ICML); 2012. p. 179–186.
[35] Sutton R, McAllester D, Singh S, et al. Policy gradient methods for reinforcement learning with function approximation. Advances in Neural Information Processing Systems (NIPS) 12; 2000. p. 1057–1063.
[36] Bhatnagar S, Sutton R, Ghavamzadeh M, et al. Natural actor-critic algorithms. Automatica. 2009;45:2471–2482.
[37] Wirth C, Fürnkranz J, Neumann G. Control of the multi agent micro-robotic platform micron. IEEE International Conference on Control Applications; 2016. p. 1414–1419.
[38] Goodfellow I, Bengio Y, Courville A. Deep learning. MIT Press; 2016.
[39] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. Nature. 2015;518:529–533.