Computing and Informatics, Vol. 42, 2023, 1158–1183, doi: 10.31577/cai_2023_5_1158

MULTI-AGENT DYNAMIC LEADER-FOLLOWER PATH PLANNING APPLIED TO THE MULTI-PURSUER MULTI-EVADER GAME

Mohammed El Habib Souidi, Makhlouf Ledmi,
Toufik Messaoud Maarouk, Abderrahim Siam,
Abdeldjalil Ledmi

Department of Computer Science, ICOSI Lab, University of Khenchela, 40004 Khenchela, Algeria
e-mail: souidi.mohammed@univ-khenchela.dz

Abstract. Multi-agent collaborative path planning focuses on how the agents have to coordinate their displacements in the environment to achieve different targets or to cover a specific zone in a minimum of time. Reinforcement learning is often used to control the agents' trajectories in the case of static or dynamic targets. In this paper, we propose a multi-agent collaborative path planning based on reinforcement learning and leader-follower principles. The main objectives of this work are the development of an applicable motion planning in a partially observable environment, and also, to improve the agents' cooperation level during the tasks' execution via the creation of a dynamic hierarchy in the pursuit groups. This dynamic hierarchy is reflected by the possibility of reattributing the roles of Leaders and Followers at each iteration in the case of mobile agents to decrease the task's execution time. The proposed approach is applied to the Multi-Pursuer Multi-Evader game in comparison with recently proposed path planning algorithms dealing with the same problem. The simulation results reflect how this approach improves the pursuit capturing time and the payoff acquisition during the pursuit.

Keywords: Multi-agent system, path planning, pursuit-evasion game, reinforcement learning

Mathematics Subject Classification 2010: 68T40-68T42

1 INTRODUCTION

Multi-agent Pursuit-Evasion Game (PEG) can be considered a multi-task problem in which different groups of pursuers' agents are trying to block the motion of another group of detected evaders' agents [1]. PEG is usually processed through the utilization of a task coordination mechanism and a path planning method. On the one hand, the task coordination mechanism [2] is used to allow an efficient formation of different multi-agent coalitions able to execute the multi-task problem. On the other hand, the path planning method [3] allows the pursuers' agents to trace the trajectories leading them to the evaders' positions.

Multi-agent organizational models [4] are considered as a type of multi-agent task coordination mechanisms. An organizational model can be defined as a meta-model reflecting the relations between the concepts used simultaneously to coordinate the collective behavior of the agents. For example, in [5] the authors used the concepts of Agent, Group, and Role and the relations between them to propose a multi-agent organizational metamodel.

In the recent research activities, MAS organizational modeling frameworks [6] are enormously used to the coordinate the tasks in the PEG. Recently in [7], the authors have based on the different concepts forming the Yet Another Multi-Agent Model (YAMAM) [8] to create an efficient pursuit groups' access mechanism. Furthermore, supervised and unsupervised machine learning methods [9] are used in combination with organizational models to improve the tasks' coordination. On the one hand in [10], the authors used the neural networks' layer [11] to extract the features of the AGRMF model. To allow the coalition of the pursuers with similar features, the extracted features are processed via a self-organizing map layer. On the other hand, in [12], the authors used K-means [13] in order to group the similar evaders characterized by the best parameters among the data set.

Multi-agent collaborative path planning can be defined as the generation of a continuous series of movements from the initial to the final state of each agent, while at the same time avoiding collisions with the other agents. Markov Decision Process (MDP) [14] is a stochastic process usually used in MAS with the aim of modeling the path environment. This modeling allows the agent to make the decision according to several possible transitions. The main goal of reinforcement learning [15] is to provide the agent with an intelligent behavior during the movements through the optimization of the expected payoffs. However, in decentralized multi-agent path planning, each agent moves without taking into consideration the behavior of other agents. In addition, the multi-agent path planning problem is processed via the use of different optimization methods such as Genetic Algorithm [16, 17], Particle Swarm Optimization (PSO) [18, 19], as well as Artificial Potential Field [20, 21].

MAS collaborative path planning is both important and challenging for several reasons. The first objective of MAS path planning is to guarantee that agents can displace without colliding with each other or with the environment's obstacles. In complex environments with multiple dynamic agents such as the PE game environment, finding collision-free paths for all agents becomes increasingly hard. Thus, collision avoidance can be considered crucial to ensure the safety as well as the integrity of the agents and their environment. Moreover, MAS path planning involves the coordination and cooperation of the agents with each other with the aim of achieving common objectives. This fact requires the agents to consider the actions and the intentions of the other agents, and to predict their future behavior. Consequently, coordinating MAS efficiently is a challenging task, knowing that the agents have to balance their individual objectives with the overall system's goals. Furthermore, we can easily note that real-world scenarios usually involve uncertainty as well as dynamic changes in the environment. In some cases, agents have incomplete information about the positions, velocities, or intentions of the other agents. In addition, the environment itself may change over time due to dynamic obstacles, unpredictable events, or varying objectives. Therefore, the incorporation of uncertainty and adaptability into MAS path planning process further increases the complexity of the problem.

In this paper, we introduce a new cooperative multi-agent path planning through its application to the PEG. This method is based on a dynamic attribution of the sub-roles Leader and Followers to the pursuers belonging to the same group according to their dynamic environmental positions to decrease the pursuit capturing time. Knowing that the environment is modeled according to MDP principles, the hierarchy of each pursuit group is dynamically updated in relation to the rewards detected by the pursuers in the pursuit environment. The main contributions of this paper can be summarized as follows:

- The proposition of a MAS collaborative path planning based on the hierarchization of the agents in dynamic roles as well as Q-learning to orient the agents in selecting their directions.
- The applicability of the proposed approach in a partial observable environment. Knowing that the followers are not required to know the targets' positions in order to perform their tasks.
- The application of the proposed approach to the Multi-Pursuer Multi-Evader Game (MPMEG) in comparison with recent path planning methods. During this study, we have taken into account the capturing time as well as the pursuers' development during the game execution in order to prove the feasibility of the proposed approach.

The paper is organized as follows: In Section 2, we discuss the main related works processing the multi-agent coordination as well as the PE problems in relation to the proposed work. In Section 3, we describe the PE game environment via the definition of its different components. Furthermore, we explain the difference between the agents existing in the environment, and how this last is modeled according to the MDP principles. Section 4 details how the PE game is processed through the application of the new path planning proposed in this paper. Section 5 reflects the simulation results obtained in comparison with a recent approach also dealing with PE problem. Finally, Section 6 highlights concluding points regarding the proposed approach.

2 RELATED WORK

Among the recent and interesting works regarding the multi-agent path planning in PEG, we note [22], in which the authors proposed a static leader-follower path planning based on reinforcement learning. This work is based on the decomposition of tasks between the pursuers. In other words, each pursuit group is composed of a set of Leaders and a set of Followers. In comparison with this path planning algorithm, our proposal is based on the attribution of only one role Leader per each pursuit group to focalize the pursuers on the global best solution. Moreover, in this work, the roles Leader and Followers of the moving pursuers belonging to each group are dynamically reattributed before each pursuit iteration according to the agents' new positions. Knowing that the dynamic reattribution of these roles positively impacts the pursuit capturing time as well as the pursuers' development in case of moving evaders.

In [23], the authors processed the PEG through the processing of the constraints linked to the environment changes during the pursuit. They proposed a deep reinforcement learning method allowing the capture of the evaders even if the number of pursuers has changed. Specifically, they have based on a deep deterministic policy gradient (DDPG) framework and bi-directional recurrent neural network (Bi-RNN) with the aim of studying the PEG in the case where the evaders are faster than the pursuers, but less numerous than them. However, the authors did not introduce a task allocation method to define which pursuers must perform the pursuit of specific evaders in the case of MPMEG. This fact negatively impacts the autonomy of the approach and its application in the real world. To overcome this limitation in this proposed approach, we have based on the MAS task coordination mechanism based on YAMAM organizational model proposed in [7].

In [7], the authors used Q-learning [24] in order to allow the pursuers to move in a decentralized way with the aim of obtaining the maximum payoffs detected in environment cells. Knowing that the payoffs are calculated in relation to the distance separating each environment cell from the cell containing the concerned evader. However, the negative point of this approach is reflected by the fact that it is only applicable in a completely observable environment. In other words, all the pursuers need to know the exact position of the concerned evader at each pursuit iteration. However, in the proposed approach, the pursuit can be applied in a partially observable environment. In other terms, only the leader needs to know the exact position of the concerned evader.

In [25], the smart pursuers undertook Watkins's Q(λ)-learning algorithm with the aim of learning from their interactions. The method the authors used is an extended version of Q-learning and eligibility traces. It utilized saved knowledge until the first occurrence of an exploration. We can note that in the two last related works [24, 25] the authors explained that each pursuer takes independent decisions regarding its action-value function and the updates of its information space. In comparison with the proposed approach in this paper, the path planning proposed in [25] only processed the Multi-Pursuer Single-Evader Game (MPSEG), which is a less complex problem in relation to the MPMEG processed in this paper.

In [26], the authors have based on game theoretic principles and Q-learning to process the PE problem. After the formation of the hunting team and via learning from the evader's path strategy, the trajectory of the evader's limited T-step cumulative payoff is generated and adjusted to the pursuer's strategy set. Also, the game theoretic Nash equilibrium solution is obtained through the resolution of the cooperative pursuit game. finally, each pursuer follows the generated equilibrium strategy to complete the pursuit task. However, this approach is based on a centralized communication method with several lacks. In other words, this approach totally depends on the virtual manager which identifies the pursuers and the evader, records the agents' paths, and selects the best solution in a centralized way in the case of finding several balanced solutions. Consequently, we can conclude that the approach is not realizable in the case where the virtual manager is out of order. In comparison with the proposed approach, the PE game processing is more distributed on the integrality of the pursuers. In other words, in case of the leader's failure, this last is immediately replaced by the pursuer with the highest pursuit skills, which will be introduced in the next section of this paper.

PEG can even be considered a clearing zone problem where the pursuers are trying to cover the pursuit environment in a minimum time to detect the evaders' positions. In [27], a Partially Observable Markov Decision Process (POMDP) algorithm is illustrated to localize the mobile target in a known graph. The main objective is to perform the capture of the targets through the clearing of the graph as quickly as possible. Otherwise, this approach is clearly limited by the pursuers' field of view as well as the camera type used in order to detect the evaders. In relation to the proposed work, the approach proposed in [27] is not based on Q-learning principles. Moreover, it processed the Single-Pursuer Single-Evader game. Finally, the PE game processed in [27] is not situated in the grid of cells environment.

The PEG is also processed via the avoidance of the different obstacles detected in the environment. In [28], the authors proposed a new obstacle avoidance path planning-based MDP framework and bug algorithms [29]. Knowing that the main objective of the bug algorithm is to unidirectionally turn around the obstacle until finding the leaving point, they proposed to find the leaving point according to the payoffs returned by the application of the MDP reward function. This approach provided interesting results in relation to the precedent bug algorithms, however, this approach requires an MDP environment modeling in order to be applicable. In the PE game, path planning algorithms and obstacle avoidance methods must be combined to allow the agents' displacement in a pursuit environment that contains obstacles. Thus, in the case of a pursuit environment with obstacles, the proposed path planning in this paper can be combined with the obstacle avoidance algorithm proposed in [28] to provide an efficient pursuers' behavior.

Regarding the task sharing between the pursuers, in [7] the authors based on the concepts of Agent, Role, Task, and Skills forming the YAMAM organizational model to develop a pursuit groups access mechanism allowing an equitable and stable grouping of the pursuers during the pursuits. In [30], they proposed a coalition formation algorithm for the student agents selecting the courses proposed by the university in E-learning system. Precisely, they introduced a voting procedure allowing the coalition of the agents and also the allocation points of the different courses. The proposed method makes the agents able to independently express their preferences and simultaneously use the information furnished by the precedent rounds to vote intelligently and strategically. In MPMEG, in addition to the path planning, a task-sharing method should also be provided to determine which pursuers have to pursue each evader. In this paper, we have based on the task-sharing method proposed in [7] to allow the pursuit groups' formation.

3 PROBLEM DESCRIPTION

PEG is considered a multi-agent complex problem in which moving agents known as Pursuers are forming different pursuit groups in order to capture other moving agents known as Evaders. The main objective of this game is to decrease the pursuers displacements during the pursuit by providing them an intelligent behavior. On the one hand, this behavior is usually reflected by the use of an intelligent task coordination algorithm allowing the pursuers to be regrouped in different pursuit groups according to their abilities. On the other hand, it is reflected by the use of collaborative path planning algorithms that provides to each pursuer an optimal trajectory to follow in order to achieve the goal location.

In this work, the PE game will be handled in a limited grid of cells environment, in which the agents can displace from a cell to another one according to the detected information as well as to their velocities. The agents can indirectly communicate through the modification of environment information. Each cell is characterized by Cartesian coordinates and also a vector containing different information. This set of information represents the expected payoff that could be obtained by the pursuers in the case of reaching the concerned cell. Moreover, it reflects the information regarding the cell occupation. Knowing that each cell can only contain one agent at a time. Moreover, the rewards contained in each cell are dynamically updated during each pursuit iteration according to the new positions of the moving evaders.

In this proposal, there exist two types of agents in the environment, the evaders and the pursuers in accordance with the PEG principles. These agents can move in four different directions: up, down, left, and right. Moreover, they are equipped with limited sensors allowing them to read the information contained in the adjacent cells. Knowing that the main objective of the PE game is to stop the movement of the detected evaders as quickly as possible, the evaders are randomly moving in the environment with the aim of avoiding their captures. Furthermore, each evader needs the coalition or a pursuit group formed by a specific number of pursuers in order to be captured.

With the aim of parallelly performing the pursuit of the detected evaders, the pursuit group regarding each evader must contain a specific number of pursuers according to the evader's requirement. In addition, the pursuers must follow a specific collaborative path planning in a centralized or decentralized way in order to move in the direction of these evaders. The centralized path planning concerns the case where the pursuers are collaborating during their displacements. However, the decentralized way regards the case where the pursuers are independently moving in the direction of the goal cell. To propose a new collaborative path planning in this paper, the PE environment is modeled as an MDP framework (S, A, R, T):

- $S = \{s_1, s_2, \ldots, s_n\}$: the set of the environment states (cells). In the PE game, the states represent the cells existing in the environment.
- $A = \{a_1, a_2, \ldots, a_n\}$: the set of actions that the agent can effectuate. In the PE game, the actions can be described by the displacement of the agent in the environment, and also by the reading of the information contained in the adjacent cells.
- $R(s, a)$: The reward function determines the payoff could be obtained by an agent if it reaches the state s through the execution of the action a in the PE game, the payoff is proportionally inverse to the distance between the cell containing a pursuer and the cell containing the concerned target.
- $T(s, a, s')$: The transition function determines the impact of an agent's action on the environment. It also determines the probability of switching from the state s to s' by executing the action a according to the payoff returned by the two states.

To allow the pursuers' collaboration, we propose to introduce two types of pursuers, the leader and the followers. The leader of a pursuit group is the detector of the concerned evader's position. In other words, the leader of a pursuit group is the pursuer occupying the closest position in relation to the concerned evader's position. The role of the leader is to guide the followers belonging to the same group to capture the evader via the modification of the environmental information. The followers must minutely move according to the path traced by the leader. The Q-learning in this proposal is used to guide the leaders to move in the direction of the pursuers and also to allow the followers to move according to the desired path traced by the leaders.

Figure 1 is a part of the pursuit environment taken from the effectuated simulations in which 2 evaders are pursued by two pursuit groups formed by 4 pursuers each. The variables of the vector $[Var_1, Var_2, Var_3, Var_4, Var_5]$ contained in each cell are explained in the following way:

- Var_1: the reward returned to the leader of the first pursuit group in the case of reaching the concerned cell.

- Var_2: the reward returned to the leader of the second pursuit group in the case of reaching the concerned cell.
- Var_3: the cell index, it can contain 2 different values:

$$Index\_cell = \begin{cases} 1, & \text{if } (\exists Ag \in agents\_list \wedge Ag \subset cell) \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

- Var_4: the reward returned to the followers of the first pursuit group in the case of reaching the concerned cell.
- Var_5: the reward returned to the followers of the second pursuit group in the case of reaching the concerned cell.

Furthermore, this figure showcases 5 dispersed agents having different colours. The green agent represents one of the detected evaders. The yellow agent represents a pursuer and at the same time the leader of the pursuit group trying to capture the green evader. The red agents are pursuers and at the same time followers of the yellow leader. The different payoffs of each cell are calculated in relation to the distance between the evader and the pursuers according to their types.

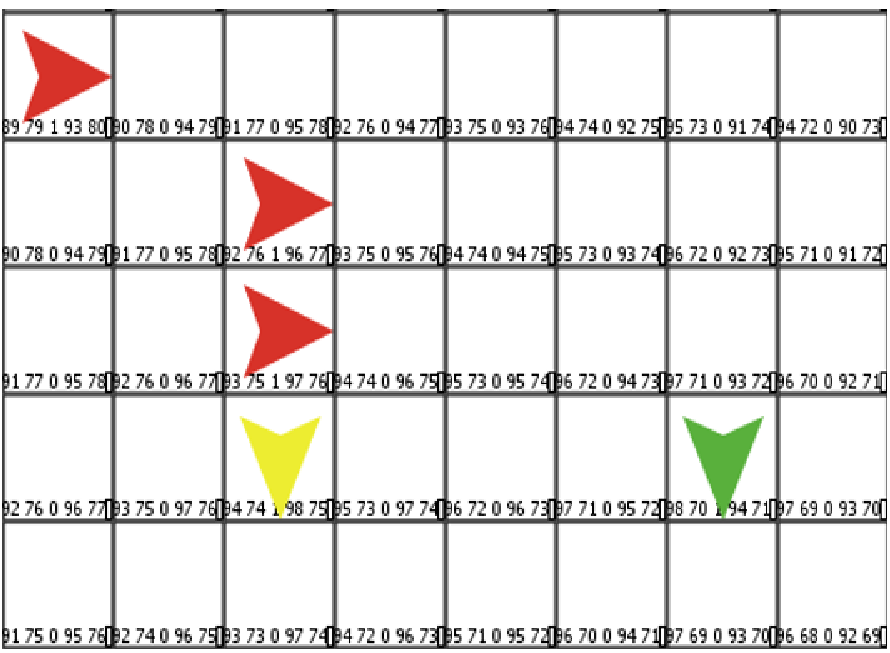

Figure 1. PEG environment's part

4 LEADER-FOLLOWER PATH PLANNING

In this section, we introduce the proposed path planning algorithm. Firstly, we explain how this algorithm can be generally used. Secondly, we detail how the algorithm is applied in order to provide the pursuers' trajectories during the PEG processing.

Algorithm 1 details how the leader-follower principle is used in order to manage the multi-agent path planning until the end of the task execution. The agent's skill can be defined as different ability factors of the agent that play an important role during the task execution. For example, in the case of the PEG, the environment position and velocity of each pursuer are considered as important factors with a high impact on the pursuit processing. The agent's skill is generally calculated in relation to agent's ability factors as follows:

$$\lambda(Ag_i) = \Omega_1 * ab_1^i + \Omega_2 * ab_2^i + \cdots + \Omega_n * ab_n^i, \tag{2}$$

- ab_k^i: the k^{th} ability factor of the agent Ag_i,
- Ω_i: the i^{th} ability factor's coefficient of the agent, $\Omega_i \in [0, 1]$.

We note that the agents' skill calculation is updated before every new iteration with the aim of reattributing the dynamic role of Follower to the best agent belonging to the group. The desired trajectory can be defined as the path taken by the leader. The desired trajectory is updated according to the leader's last position. In MDP environment, this update is performed through the modification of followers' expected payoffs in the environment, as shown in Figure 1. The role of the followers is to move according to the desired trajectory traced by the leader. The *leader-index* variable shown in Algorithm 1 allows the attribution of the role Leader to only one agent.

Algorithm 2 details how the leader-follower path planning is used in order to solve the PE game. The first step of the PE game is the detection of Cartesian coordinates of the existing evaders. To do this, each agent scans a specific surface of the environment. A pursuit group for each evader will be created to stop the movement of this last. With the aim of optimizing the access to the created pursuit groups, we have based on a recent task coordination mechanism based on Yet Another Multi-Agent Model (YAMAM) [7]. This organizational modeling framework is based on 4 concepts: Agent, Role, Task, and Skills. In order to participate in the pursuit achievement (task), the agent must obtain the role of Pursuer proposed by a specific group. To get this role, the concerned agent must be characterized by specific skills' degrees.

After the pursuers' roles attribution, the leader and the followers of each pursuit group will be designated. Knowing that in each pursuit group only one pursuer can obtain the role of Leader. In order to play this role, the pursuer must be characterized by the highest pursuer' skill degree. In relation to PEG, this degree is calculated in relation to the reward of the pursuer's position, the pursuer's velocity,

```
Input: task-execution = false;
Agents-group-initialization();
while task-execution() = false do
    leader-index ⟵ false;
    Agents-skills-calculation();
    for each agent_i do
        if agent_i − skills = max ∧ leader-index = false then
            Role-attribution(agent_i, Leader);
            leader-index ⟵ true;
        end
    end
    Leader-action();
    Update(desired-trajectory());
    for each agent_i do
        if agent_i ≠ Leader then
            Role-attribution(agent_i, Follower);
            Follow-desired-trajectory(agent_i);
        end
    end
    Check(task-execution());
end
Output: task-execution = true;
```

Algorithm 1: Multi-agent leader-follower path planning

as well as the sensor's length of the pursuer. The agent's skill is calculated as follows:

$$\lambda(P_i) = \alpha * r_i + \beta * v_i + \gamma * SL_i, \tag{3}$$

where:

- r_i: the reward of the pursuer$_i$ according to its position in the environment;
- v_i: the agent's velocity determining the number of cells that the pursuer$_i$ can cross during an iteration;
- SL_i: represents the average sensor's length of the pursuer$_i$.
- α: the reward coefficient, $\alpha \in [0, 1]$;
- β: the velocity coefficient, $\beta \in [0, 1]$;
- γ: the Sensor's Length coefficient, $\gamma \in [0, 1]$.

Knowing that the pursuer can move in 4 different directions, the average Sensor's Length is calculated as follows:

$$SL_i = \gamma_1 * SL_i^{up} + \gamma_2 * SL_i^{down} + \gamma_3 * SL_i^{left} + \gamma_4 * SL_i^{right}. \tag{4}$$

In the case where the role of Leader is already attributed, the pursuer automatically obtains the role Follower. The role Leader consists on moving forward in the direction of the evader according to the leader's payoff shown in Figure 1. After the leader movement, the followers expected payoffs are updated. The role of the followers is to move forward in the direction of their leader according to the followers' payoff shown in Figure 1.

Knowing that the environment is modeled according to MDP principles, the Q-learning algorithm can be placed within MDP framework with the aim of allowing the pursuers to move via the learning of the optimal Q-values defined in Equation (5). This reinforcement learning method allows to learn a strategy, which indicates what action to perform in each state of the system. It works by learning a state-action value function denoted Q which makes it possible to determine the potential gain. In other words, in Q-learning, each pursuer executes an action (a) in relation to the state (s) and to the function Q. The pursuer then perceives the new state (s') and a reward (r) from the environment before the update of the Q function.

$$Q^*(s,a) = R(s,a) + \Lambda \sum_{s'} T(s'|s,a) V^*(s'), \tag{5}$$

where:

$$V^*(s) = \max_a Q^*(s,a) \tag{6}$$

and

- Λ: The discount factor ($\Lambda \in [0,1]$).

During every iteration, the pursuers takes the decision (moving up, down, right, or left) that maximizes their payoff according to the equation below:

$$Q_{i+1}(s,a) = Q_i(s,a) + \alpha_i * [r_{i+1} + \Lambda * \max_{a'}(Q_i(s_{Up},a'), Q_i(s_{Down},a'), Q_i(s_{Left},a'), Q_i(s_{Right},a')) - Q_i(s,a)], \tag{7}$$

where

- α_i: The step-size sequence.

In the case of a leader path planning, the reward r of the Equation (7) is calculated in relation to the distance separating the leader from the concerned evader as follows:

$$r_{Leader} = r_{max} - \sqrt{(CC_{Leader_x} - CC_{Evader_x})^2 + (CC_{Leader_y} - CC_{Evader_y})^2}. \tag{8}$$

In the case of a follower path planning, the reward r is calculated in relation to the distance separating the follower from the leader of the pursuit group they belong to:

$$r_{Follower} = r_{Leader} - \sqrt{(CC_{Leader_x} - CC_{Follower_x})^2 + (CC_{Leader_y} - CC_{Follower_y})^2}, \tag{9}$$

where:

- r_{max}: the reward could be returned to the pursuer in the case of reaching one of the concerned evader's adjacent cells;
- $(CC_{Leader_x}, CC_{Leader_y})$: Leader Cartesian coordinates;
- $(CC_{Follower_x}, CC_{Follower_y})$: Follower Cartesian coordinates;
- $(CC_{Evader_x}, CC_{Evader_y})$: Evader Cartesian coordinates.

Knowing that before the displacement of the pursuers, the evaders randomly move (random direction) in the environment according to their velocities, as shown in Algorithm 2. The pursuit iteration can be defined as the execution of a possible transition by each agent. In other words, it regards the displacement of the agents from their actual cells to one of their adjacent cells. Before any pursuit iteration, the roles Leader and Followers of each pursuit group are updated to optimize the capturing time and the payoff acquisition. This update is due to the fact that the pursued evaders change their positions in the environment during each pursuit iteration by moving from one cell to another according to a specific velocity. This update is effectuated according to the dynamic distance separating each pursuer from the new position of the concerned evader.

The PEG is considered over when the average payoff obtained by the pursuers reaches a specific value. This value is represented by the variable *max-payoff* in Algorithm 2. Max-reward can be initialized by a value related to the studied case. The index of the pursuit is verified as follows:

$$IC = \begin{cases} \text{True}, & \text{if } \left(\frac{r_{P_1}+r_{P_2}+\cdots+r_{P_{np}}}{np} = r_{max}\right), \\ \text{False}, & \text{otherwise}, \end{cases} \tag{10}$$

where:

- IC: the index of the pursuit capture;
- np: the number of existing pursuers.

Scalability in a MAS refers to the system's ability to manage an increasing number of agents without significant degradation in performance or efficiency. It involves designing the system in a way that allows it to accommodate larger agent populations and more complex interactions without sacrificing its functionality or responsiveness. As shown in Algorithm 2, each pursuer calculates its pursuit skill independently of the other pursuers. This fact proves that the proposed algorithm is based on a decentralized calculation during the selection of the groups' leaders. Therefore, we can conclude that the increase of the agents' number does not negatively impact the algorithm performance.

5 SIMULATION RESULTS

This part summarizes the main simulations performed to showcase the efficiency and also, to prove the feasibility of the approach proposed in this paper. To do

```
Input: Index-capture = false;
 Evaders-detection();
 Broadcast(Evaders-coordinates);
 Pursuit-groups-formation();
 Distance-calculation();
 for each Agent_i do
 |  Role-Atrribution (Agent_i);
end
Initialize(max-payoff);
 while Index-capture = False do
 |  R ⟵ 0;
 |   Evaders-move(random);
 |   Update(r_leader);
 |   for each Group_k do
 |   |  for each P_i do
 |   |  |  if P_i ∈ Group_k = true then
 |   |  |  |  Distance-calculation (P_i , E_k);
 |   |  |  |   λ(P_i);
 |   |  |  |   list_k[] ⟵ add(λ(P_i));
 |   |  |  end
 |   |  end
 |   |  for each P_i do
 |   |  |  if condiλ(P_i) = max(list_k[]) then
 |   |  |  |  Leader_k ⟵ P_i;
 |   |  |  |   Move-to(P_i, E_k);
 |   |  |  |   Update(r_follower);
 |   |  |  else
 |   |  |  |  Follower_k[] ⟵ add(P_i);
 |   |  |  end
 |   |  end
 |  end
 |  for each Group_k do
 |  |  for each P_i do
 |  |  |  if P_i ∈ Follower_k[] = true then
 |  |  |  |  Move-to(P_i, Leader_k);
 |  |  |  end
 |  |  |  R ⟵ R + r_{P_i};
 |  |  end
 |  end
 |  if R/np = max-payoff then
 |  |  Index-capture ⟵ true;
 |  end
end
Output: Index-capture = true;
```

Algorithm 2: Pursuit-evasion based on leader-follower path planning these, we have used NetLogo 5.0.4 [31], which is an open-source oriented agent platform. It is based on two kinds of agents: the patches and turtles. The patches are situated agents which can stock dynamic information. We have used this type of agent with the aim of implementing our grid of cells environment. Specifically, each patch represents a cell containing the dynamic payoffs. The turtles are the mobile agents which can move from one patch to another. This second type is used to simulate the behaviour of our pursuers (leaders and followers) as well as our evaders. The experiments will be handled in a limitary 100×100 grid of cells environment, where 8 pursuers are trying to capture 2 evaders. Knowing that each evader requires a pursuit group formed by 4 pursuers to be captured. Regarding, the agents' sensor length (SL), we have equipped each agent with sensors able to obtain the information contained in each adjacent cell (Up$_{cell}$, Down$_{cell}$, Left$_{cell}$, Right$_{cell}$). The initial agents' positions are detailed in Table 1.

Agent	Initial Cartesian Coordinates
P_1	(50, 75)
P_2	(75, 75)
P_3	(25, 50)
P_4	(50, 50)
P_5	(75, 50)
P_6	(37, 38)
P_7	(62, 37)
P_8	(50, 25)
E_1	(55, 55)
E_2	(44, 44)

Table 1. The agents' initial positions

In order to focalize the simulation studies only on the impact of the path planning on the PEG, we have based on the following pursuit groups generated through the application of the task coordination mechanism proposed in [7] with the aim of capturing the mobile evaders E_1 and E_2:

$$\textit{Pursuit-Group}(1) = \{P_4, P_5, P_1, P_2\},$$

$$\textit{Pursuit-Group}(2) = \{P_6, P_3, P_8, P_7\}.$$

In order to showcase the improvement brought by the proposed path planning, we have seen the usefulness of comparing it with two recent path planning methods treating the PE game. The main difference between the compared cases is detailed as follows:

Case A: Pursuit-evasion game based on the new leader-follower path planning explained in Section 3.

Case B: Pursuit-evasion game based on the static leader-follower path planning proposed in [22] and detailed in Section 2 of this paper.

Case C: Pursuit-evasion game based on the path planning proposed in [7], where the pursuers are moving independently of each other according to the dynamic rewards detected in the environment. Specifically, there are no specific roles (Leader and Followers) regarding the pursuers. Each pursuer leads itself to perform the pursuit.

We have seen the usefulness of comparing it with the proposed approach to the cases B [22] and C [7] for the following reasons:

- In relation to the proposed approach in this paper, cases B and C are also based on reinforcement learning principles,
- Cases B and C were recently applied to the MPMEG, which is the processed PE game in this paper.
- These two approaches can also be applied in a grid of cells PE game environment used in this paper to reflect the impact of the proposed approach.

Figure 2 reflects the pursuit capturing time obtained after 20 pursuit episodes. A pursuit episode starts by the coalition formation of the pursuit groups, and ends after the capture of the different evaders. The average capturing time in the case A decreases by 9.91 % in relation to case B and 16.41 % in relation to case C. This fact is totally due to efficiency of the leader-follower proposed technic. In other words, the dynamic attribution of the sub-roles of Leader and Follower increases the goal orientation of the pursuers.

In order to prove the significance of the obtained capturing time reflected in Figure 2, we have performed the Friedman test. According to the obtained results ($X_r^2 = 27.925$), we can conclude that the obtained result is significant at $p < 0.05$ (significance level).

Figure 3 reflects the average reward obtained per iteration by the pursuers during a complete pursuit episode in the 3 compared cases. From this figure, we can note that the average reward increases in case A by 7.1 % in relation to case B, and 13.5 % in relation to case C. We can justify the flagrant difference between case A and C by the close grouping as well as the close displacement of the pursuers during the pursuit provided by the leader-follower principle. The average reward obtained per iteration (Ar_LD) in the cases A and B is calculated as follows:

$$Ar_{LD} = \sum_{i=1}^{nl} \left(r^t_{Leader_i} - r^{t-1}_{Leader_i} \right) + \sum_{i=1}^{nf} \left(r^t_{Follower_i} - r^{t-1}_{Follower_i} \right), \tag{11}$$

where

- t: the index of the pursuit iteration;
- nl: represents the number of leaders used in the game. In these simulations, we are using 2 leaders;
- nf: represents the number of followers used in the game. In these simulations, we are using 6 followers.

However, in the case of decentralized path planning (case C) in which each pursuer behaves as a leader by independently moving to the target, Ar_D is calculated as follows:

$$Ar_D = \sum_{i=1}^{np} \left(r^t_{Pursuer_i} - r^{t-1}_{Pursuer_i} \right), \tag{12}$$

where

- np: represents the number of pursuers used in the game. In these simulations, we are using 8 pursuers.

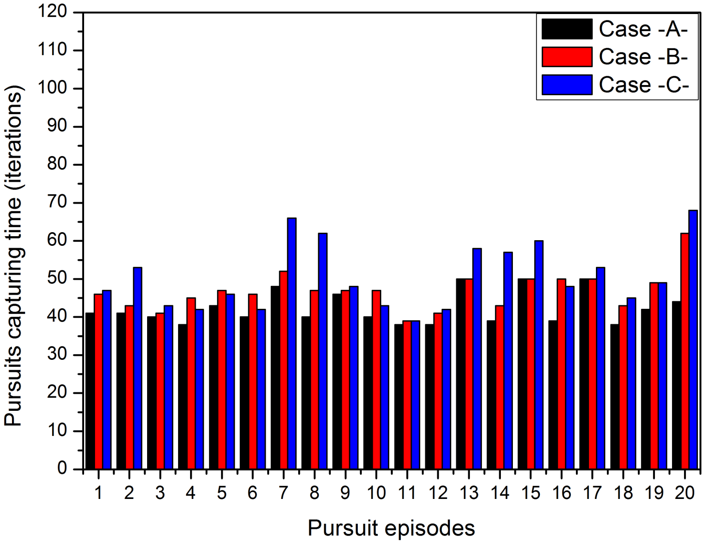

Figure 2. PEG capturing time obtained (Pursuers' motion speed = Evaders' motion speed)

In order to study the behavior of the pursuers during the pursuit, we have focused on their dynamism degree regarding the roles' attribution. This dynamism (Dy) concerns the roles' changes between the roles of Leaders and Followers in the two pursuit groups studied. In other words, if a pursuer changes its role during a pursuit iteration, then the dynamism degree is automatically incremented. It is calculated at each pursuit iteration as follows:

$$\text{For each } P_i, Dy = \begin{cases} Dy, & \text{if role}^{t-1}_{P_i} = \text{role}^t_{P_i}, \\ Dy + 1, & \text{otherwise.} \end{cases} \tag{13}$$

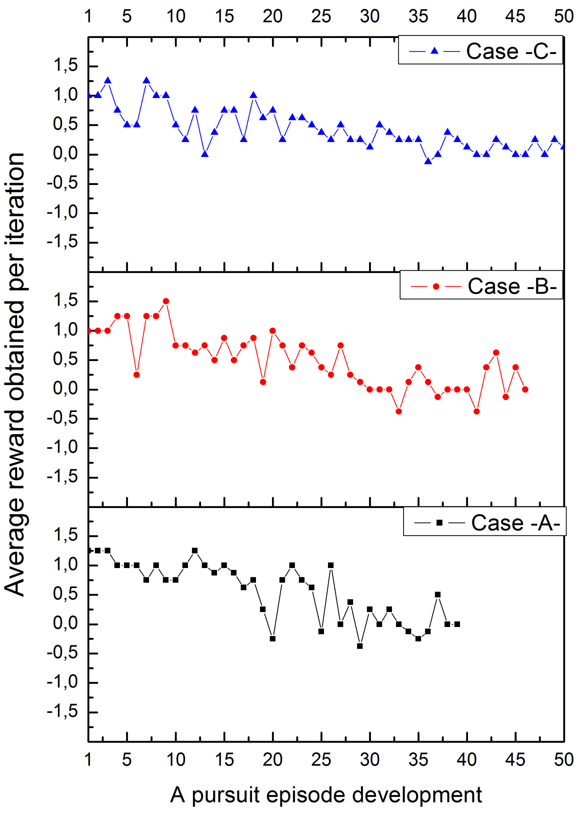

Figure 3. Pursuer's average rewards obtained per pursuit iteration (Pursuers' motion speed = Evaders' motion speed)

As shown in Figure 4, we note 8 roles attributions at the first iteration which can be explained by the creation of 2 pursuit groups in which each group is formed of 4 pursuers. Moreover, we can note that this degree increases in the case when the pursuers are closer to the evaders (from the 27^{th} iteration).

Furthermore, we have studied the dynamism of the role's attributions during 20 pursuit episodes regarding case A, as shown in Figure 5. The average dynamism obtained is 25.65. We can note that this degree is proportionally inverse to the pursuit capturing time. Knowing that the difference between the results obtained in each pursuit episode is related to the fact that the pursuers are randomly moving in the environment. Consequently, we conclude that the PEG processed in these simulations is non-deterministic.

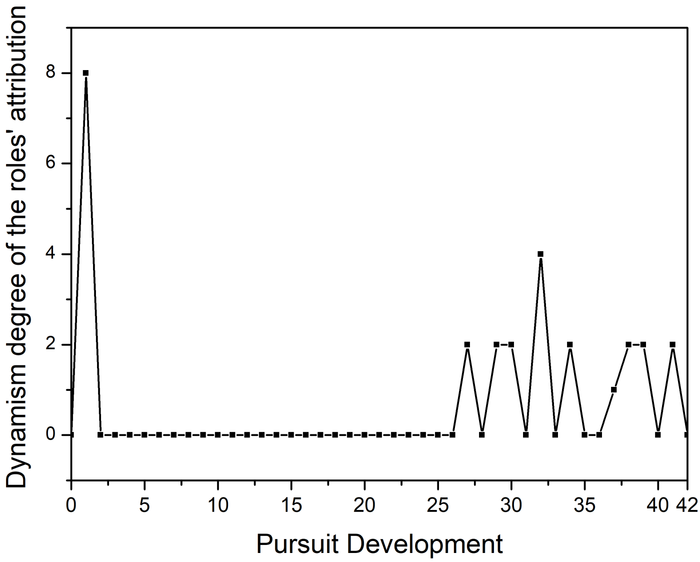

Figure 4. The study of the dynamism degree of the roles' attribution during the pursuit iterations of a complete pursuit episode

With the aim of varying the studied cases, we have doubled the evaders' motion speed in relation to the pursuers. The results shown in Figure 6 showcase the capturing times obtained after 20 pursuit episodes in the both cases. We can easily note that the difference between the three cases increases in relation to the case using the same motion speed (Figure 2). The average capturing time in case A decreases by 18.14 % comparing with case B and 33.81 % comparing with case C. We can conclude that the difference in the capturing time between the two cases increases in relation to the increase of motion speed's difference between the pursuers and evaders.

In accordance to the reflected results in Figure 7, we observe that the maximum average reward is reached after 49 pursuit iterations in case A, 56 iterations in case B, and 76 iterations in case C. In relation to the path planning proposed in case B and C, the dynamic leader-follower approach increases the cooperation level between the pursuers belonging to the same group in order to efficiently accomplish the assigned task.

Moreover, we have also effectuated the Friedman test on the development of the payoff acquisition shown in Figure 7. Knowing that the Friedman test requires the same number of values to be performed, we have only taken into account the results obtained in the first 49 pursuit iterations. According to the obtained result

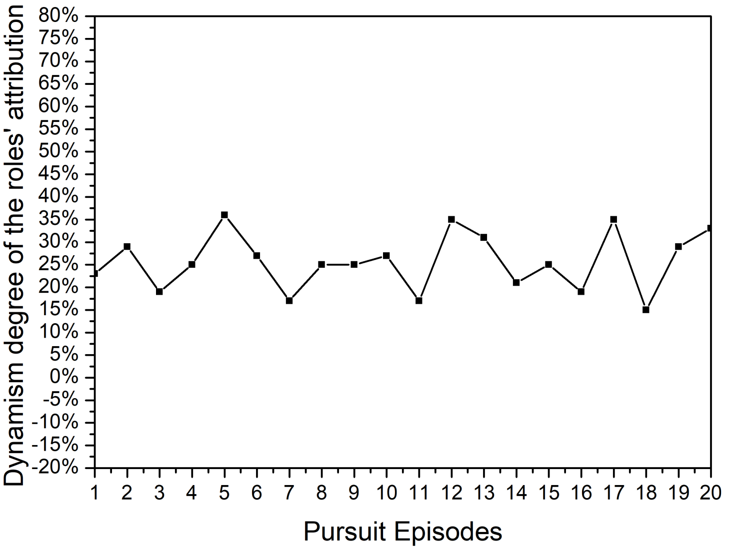

Figure 5. The study of the dynamism degree of the roles' attribution during 20 pursuit episodes

($X_r^2 = 25.5408$), we can conclude that the obtained result is significant at $p < 0.05$.

In order to compare the dynamism of the roles' attribution between the cases with different evaders' velocities, we have also studied this factor during 20 pursuit episodes in which the velocity of the evaders is doubled in comparison with the pursuers' velocity, as shown in Figure 8. From the showcased results, we can note that the average dynamism degree is 39.9. In comparison with the results reflected in Figure 5, we deduce that the dynamism degree increases by 35.71 %. From this result, we can conclude that the dynamism of the roles' attribution increases in relation to the increase of the evaders' velocity. Knowing that when the pursuers change their roles, that means that they find a better strategy in relation to the undertaken strategy during the pursuit iteration $(t-1)$.

Table 2 summarizes the average capturing time obtained after 20 pursuit episodes and also the average reward obtained per iteration in the case where the speed of the pursuers and evaders is the same, and also in the case where the evaders' motion velocity is doubled.

In this section, we have studied the proposed path planning algorithm in comparison with recent approaches [7, 22] dealing with the same problem. During these simulations, we have varied the evaders' velocities to increase the complexity of the pursuit. From the obtained results, we have constated that the pro-

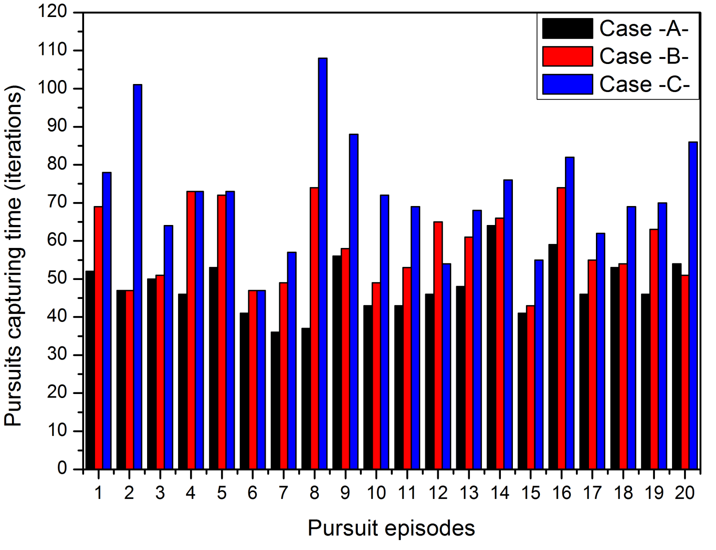

Figure 6. PEG capturing time obtained (2 × pursuers' motion speed = evaders' motion speed)

posed approach improves the capturing time as well as pursuers' reward acquisition in relation to the compared methods. Furthermore, we have noted that the proposed approach is less impacted by the increase of evaders' velocity in comparison to the other works, which is due to the goal-orientation of the novel approach. Also, we have studied the pursuit groups reorganization through the dy-

		Case A	Case B [22]	Case C [7]
Pursuers' motion speed = Evaders' motion speed	Average capturing time (Iterations)	42.25	46.9	50.55
	Average reward obtained per iteration	0.56	0.489	0.425
2 × Pursuers' motion speed = Evaders' motion speed	Average capturing time (Iterations)	48.05	58.7	72.6
	Average reward obtained per iteration	0.466	0.406	0.276

Table 2. Simulation results

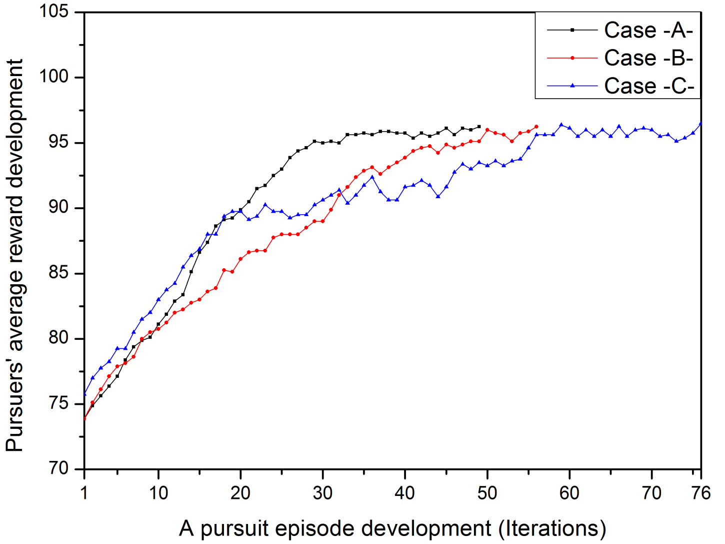

Figure 7. Pursuer's payoffs development during a complete pursuit (2 × pursuers' motion speed = evaders' motion speed)

namic reattribution of the roles (leader and followers). From this last study, we have concluded that the reorganization degree of the pursuit groups increases in relation to pursuers velocity. However, it is proportionally inverse to the capturing time.

6 CONCLUSIONS

In this paper, we proposed a new multi-agent path planning based reinforcement learning and leader-follower principles. The main objective of this work is to increase the collaboration level between the agents during the tasks' execution. In addition, this approach includes a certain dynamism regarding the roles of Leader and Follower according to the environment changes. Knowing that the principle of the followers is to follow the leader's path, the proposed path could be also used in a partially observable environment. In other words, the followers move to capture the evaders without knowing their exact positions. To reflect the feasibility of this work, we applied it to the PE game in comparison with recent path planning approaches. The simulation results proves that the new path planning improves the pursuit capturing time and also pursuers' payoff development during the pursuit. Furthermore, we have processed the PEG in the case where the evaders' velocity is superior to that

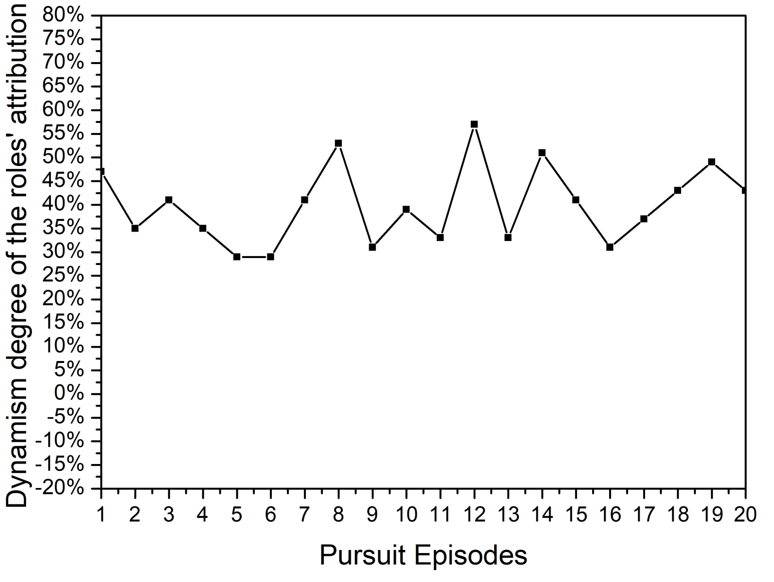

Figure 8. The study of the dynamism degree of the roles' attribution during 20 pursuit episodes (2 × pursuers' motion speed = evaders' motion speed)

of the pursuers. From this study, we have concluded that the dynamism of the pursuit groups increases in relation to the increase of the evaders' velocity. Regarding the pursuit capturing time, we can constate that the new proposed approach is less affected by the increase of the evaders' velocity in comparison with the other approaches.

On the other hand, we can constate that the proposed approach does not take into consideration the complex obstacles' processing. Therefore, in the future work, we will equip this path planning algorithm with a new obstacle avoidance method that takes into account the processing of complex and dynamic obstacles.

Regarding the real-world application, the proposed path planning algorithm can easily be applied to control the cooperative behavior of mobile robots or UAVs during the processing of complex tasks, such as warehouse automation or targets' capture. However, in case of UAVs or robots, these lasts must be equipped with a communicating system allowing them to calculate the distance between each other. Moreover, these entities must have access to some information about a part of each other (agents belonging to the same group), such as, their environment positions as well as their velocities.

REFERENCES

[1] Kuo, J. Y.—Yu, H. F.—Liu, K. F. R.—Lee, F. W.: Multiagent Cooperative Learning Strategies for Pursuit-Evasion Games. Mathematical Problems in Engineering, Vol. 2015, 2015, Art. No. 964871, doi: 10.1155/2015/964871.

[2] Durfee, E. H.: Scaling Up Agent Coordination Strategies. Computer, Vol. 34, 2001, No. 7, pp. 39–46, doi: 10.1109/2.933502.

[3] Lihui, Z.: On the Techniques of Multi-Agent Path Planning and the Collaboration. 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), IEEE, Vol. 13, 2010, pp. 430–433, doi: 10.1109/ICCASM.2010.5622835.

[4] Horling, B.—Lesser, V.: A Survey of Multi-Agent Organizational Paradigms. The Knowledge Engineering Review, Vol. 19, 2004, No. 4, pp. 281–316, doi: 10.1017/S0269888905000317.

[5] Ferber, J.—Gutknecht, O.—Michel, F.: From Agents to Organizations: An Organizational View of Multi-Agent Systems. In: Giorgini, P., Müller, J. P., Odell, J. (Eds.): Agent-Oriented Software Engineering IV (AOSE 2003). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2935, 2003, pp. 214–230, doi: 10.1007/978-3-540-24620-6_15.

[6] Dignum, V.—Aldewereld, H.—Dignum, F.: On the Engineering of Multi Agent Organizations. Proceedings of the 12th International Workshop on Agent-Oriented Software Engineering, 2011, pp. 53–65.

[7] El Habib Souidi, M.—Siam, A.—Pei, Z.—Piao, S.: Multi-Agent Pursuit-Evasion Game Based on Organizational Architecture. Journal of Computing and Information Technology, Vol. 27, 2019, No. 1, pp. 1–11, doi: 10.20532/cit.2019.1004318.

[8] Savall, M.—Pécuchet, J. P.—Chaignaud, N.—Itmi, M.: YAMAM – Un Modèle d'Organisation pour les Systèmes Multi-Agents. Implémentation dans la Plate-Forme Phœnix. Proceedings of 3ème Conférence Francophone de MOdélisation et SIMulation (MOSIM '01), 2001, pp. 25–27 (in French).

[9] Khalil, K. M.—Abdel-Aziz, M.—Nazmy, T. T.—Salem, A. B. M.: Machine Learning Algorithms for Multi-Agent Systems. Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication (IPAC '15), 2015, doi: 10.1145/2816839.2816925.

[10] Pei, Z.—Piao, S.—El Habib Souidi, M.—Qadir, M. Z.—Li, G.: Coalition Formation for Multi-Agent Pursuit Based on Neural Network. Journal of Intelligent and Robotic Systems, Vol. 95, 2019, No. 3, pp. 887–899, doi: 10.1007/s10846-018-0893-6.

[11] Abiodun, O. I.—Jantan, A.—Omolara, A. E.—Dada, K. V.—Mohamed, N. A.—Arshad, H.: State-of-the-Art in Artificial Neural Network Applications: A Survey. Heliyon, Vol. 4, 2018, No. 11, Art. No. e00938, doi: 10.1016/j.heliyon.2018.e00938.

[12] Qadir, M. Z.—Piao, S.—Jiang, H.—Souidi, M. E. H.: A Novel Approach for Multi-Agent Cooperative Pursuit to Capture Grouped Evaders. The Journal of Supercomputing, Vol. 76, 2020, No. 5, pp. 3416–3426, doi: 10.1007/s11227-018-2591-3.

[13] Lin, P.—Wang, Y.—Qi, H.—Hong, Y.: Distributed Consensus-Based K-Means Algorithm in Switching Multi-Agent Networks. Journal of Systems Science and Complexity, Vol. 31, 2018, No. 5, pp. 1128–1145, doi: 10.1007/s11424-018-7102-3.

[14] Avrachenkov, K.—Cottatellucci, L.—Maggi, L.: Cooperative Markov Decision Processes: Time Consistency, Greedy Players Satisfaction, and Cooperation Maintenance. International Journal of Game Theory, Vol. 42, 2013, No. 1, pp. 239–262, doi: 10.1007/s00182-012-0343-9.

[15] Buşoniu, L.—Babuška, R.—De Schutter, B.: Multi-Agent Reinforcement Learning: An Overview. In: Srinivasan, D., Jain, L. C. (Eds.): Innovations in Multi-Agent Systems and Applications - 1. Springer, Berlin, Heidelberg, Studies in Computational Intelligence, Vol. 310, 2010, pp. 183–221, doi: 10.1007/978-3-642-14435-6_7.

[16] de Lima, J. V. C. F.—Belo, E. M.—da Silva Marques, V. A.: Multi-Agent Path Planning with Nonlinear Restrictions. Evolutionary Intelligence, Vol. 14, 2021, No. 1, pp. 191–201, doi: 10.1007/s12065-020-00534-1.

[17] Rong, T. X.—Mei, X. H.: Robot Path Planning Based on A* Algorithm and Genetic Algorithm. 2020 International Conference on Computer Engineering and Intelligent Control (ICCEIC), IEEE, 2020, pp. 101–105, doi: 10.1109/ICCEIC51584.2020.00028.

[18] Abhishek, B.—Ranjit, S.—Shankar, T.—Eappen, G.—Sivasankar, P.—Rajesh, A.: Hybrid PSO-HSA and PSO-GA Algorithm for 3D Path Planning in Autonomous UAVs. SN Applied Sciences, Vol. 2, 2020, No. 11, Art. No. 1805, doi: 10.1007/s42452-020-03498-0.

[19] Biswas, S.—Anavatti, S. G.—Garratt, M. A.: Obstacle Avoidance for Multi-Agent Path Planning Based on Vectorized Particle Swarm Optimization. In: Leu, G., Singh, H. K., Elsayed, S. (Eds.): Intelligent and Evolutionary Systems. Springer, Cham, Proceedings in Adaptation, Learning and Optimization, Vol. 8, 2017, pp. 61–74, doi: 10.1007/978-3-319-49049-6_5.

[20] Zheng, Y.—Li, B.—An, D.—Li, N.: A Multi-Agent Path Planning Algorithm Based on Hierarchical Reinforcement Learning and Artificial Potential Field. 2015 11th International Conference on Natural Computation (ICNC), 2015, pp. 363–369.

[21] Asil, Ü.—Uslu, E.: Potential Field Path Planning and Potential Field Path Tracking. 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), IEEE, 2020, pp. 1–4, doi: 10.1109/ASYU50717.2020.9259808.

[22] Zhu, Z. Y.—Liu, C. L.: A Novel Method Combining Leader-Following Control and Reinforcement Learning for Pursuit Evasion Games of Multi-Agent Systems. 2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV), IEEE, 2020, pp. 166–171, doi: 10.1109/ICARCV50220.2020.9305441.

[23] Xu, L.—Hu, B.—Guan, Z.—Cheng, X.—Li, T.—Xiao, J.: Multi-Agent Deep Reinforcement Learning for Pursuit-Evasion Game Scalability. In: Jia, Y., Du, J., Zhang, W. (Eds.): Proceedings of 2019 Chinese Intelligent Systems Conference (CISC 2019). Springer, Singapore, Lecture Notes in Electrical Engineering, Vol. 592, 2019, pp. 658–669, doi: 10.1007/978-981-32-9682-4_69.

[24] Hwang, K. S.—Chen, Y. J.—Jiang, W. C.—Lin, T. F.: Continuous Action Generation of Q-Learning in Multi-Agent Cooperation. Asian Journal of Control, Vol. 15,

2013, No. 4, pp. 1011–1020, doi: 10.1002/asjc.614.

[25] Bilgin, A. T.—Kadioglu-Urtis, E.: An Approach to Multi-Agent Pursuit Evasion Games Using Reinforcement Learning. 2015 International Conference on Advanced Robotics (ICAR), IEEE, 2015, pp. 164–169, doi: 10.1109/ICAR.2015.7251450.

[26] Zheng, Y.—Fan, W.—Han, M.: Research on Multi-Agent Collaborative Hunting Algorithm Based on Game Theory and Q-Learning for a Single Escaper. Journal of Intelligent and Fuzzy Systems, Vol. 40, 2021, No. 1, pp. 205–219, doi: 10.3233/JIFS-191222.

[27] Ehsan, E.—Kunwar, F.: Probabilistic Search and Pursuit Evasion on a Graph. Transactions on Machine Learning and Artificial Intelligence, Vol. 3, 2015, No. 3, pp. 56–65, doi: 10.14738/tmlai.33.1313.

[28] Souidi, M. E. H.—Piao, S.—Li, G.: Mobile Agents Path Planning Based on an Extension of Bug-Algorithms and Applied to the Pursuit-Evasion Game. Web Intelligence, Vol. 15, 2017, No. 4, pp. 325–334, doi: 10.3233/WEB-170369.

[29] Sezer, V.—Gokasan, M.: A Novel Obstacle Avoidance Algorithm: "Follow the Gap Method". Robotics and Autonomous Systems, Vol. 60, 2012, No. 9, pp. 1123–1134, doi: 10.1016/j.robot.2012.05.021.

[30] Basak, S.—Mazumdar, B. D.: Multi-Agent Coalition Formation for Course Selection Strategies in E-Learning System. International Journal of Mathematical Trends and Technology (IJMTT), Vol. 6, 2014, No. 1, pp. 36–43, doi: 10.14445/22315373/IJMTT-V6P503.

[31] Tisue, S.—Wilensky, U.: NetLogo: A Simple Environment for Modeling Complexity. Proceedings of the Fifth International Conference on Complex Systems (ICCS 2004), 2004, pp. 16–21.

Mohammed El Habib Souidi is Lecturer in the Department of Mathematics and Computer Science, Faculty of Sciences and Technology, University of Khenchela, Algeria. He received his Ph.D. in computer science from the Harbin Institute of Technology (China), in 2017. His main areas of research include multiagent task coordination, reinforcement learning, game theory and path planning.

Makhlouf Ledmi received his Ph.D. in computer science from University of Batna 2, Algeria, in 2020. He is currently Associate Professor at the Mathematics and Computer Science Department at University of Khenchela and member of Knowledge Management Group ICOSI Lab. His research interests include data mining, soft computing, and bioinformatics.

Toufik Messaoud Maarouk is Lecturer in the Department of Mathematics and Computer Science, Faculty of Sciences and Technology, University of Khenchela, Algeria. He received his Ph.D. in computer science from the Constantine University, Algeria, in 2012. His main areas of research include formal methods, concurrency theory, formal semantics and distributed computing.

Abderrahim Siam received his Ph.D. in computer science from the University of Constantine, Algeria. He is working as Professor in the Department of Mathematics and Computer Science in University of Khenchela, Algeria. He is currently the Vice Rector of University of Khenchela, Algeria. His research interests include software engineering, fuzzy logics, formal methods, multiagent systems and complex systems.

Abdeldjalil Ledmi received his Engineer's degree in computer science from the University of Khenchela, Algeria in 2010. He also received his Master's degree in computer science from the same university in 2014, and his Ph.D. in computer science from the Larbi Tebessi University, Algeria in 2019. He is working as Lecturer in the Department of Mathematics and Computer Science at the University of Khenchela, Algeria. Moreover, he is affiliated as a Researcher in ICOSI Lab.