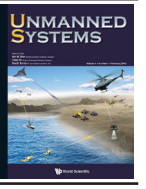
Unmanned Systems, Vol. 14, No. 2 (2026) 357–367

DOI: 10.1142/S2301385026500044

UNMANNED SYSTEMS
https://www.worldscientific.com/worldscinet/us



Development of a Centralized Conflict Free Greedy Assignment Learning Spiking Neural Network for Solving a Perimeter Defense Problem

Mohammed Thousif *,‡, Shridhar Velhal †,§, Suresh Sundaram *,¶, Narasimhan Sundararajan *,‖

*Department of Aerospace Engineering, Indian Institute of Science, Bengaluru, India

†Department of Computer, Electrical and Space Engineering, Luleå University of Technology, 971 87 Luleå Sweden

In this paper, a Centralized Conflict-free Assignment Learning Spiking neural network (C2ALS) is formulated for solving a Perimeter Defense Problem (PDP). Here, the region between the perimeter and the sensing range of the defender is divided into two layers. The outermost layer, in which the defender can only sense the intruders, is termed the sensing layer. The layer closest to the perimeter in which the defenders can sense and capture the intruders is termed the capture layer. Both layers are further divided into angular segments with respect to the center of the area to be protected. The spatiotemporal movements of both the defenders and intruders in these segments are converted into spikes and given as input to a Spiking Neural Network (SNN). The SNN is trained in a supervised manner to learn the intruder capture assignments of a defender in terms of these segments. Here, the desired assignments required for training a defender are referred to as greedy assignments because while generating them, priority is given to those segments that are relatively closer to the defender's actual location. Inhibitory connections are used in the SNN architecture to obtain assignments without conflicts. Detailed performance study results show that the proposed C2ALS approach outperforms the other existing centralized optimization-based solutions for PDP by a 17% increase in the success rates of capturing the intruders. Also, C2ALS is the first centralized spike-based learning solution for PDP, capable of generating conflict-free assignments while ensuring robust intruder assignments with minimal defender movement.

Keywords: Perimeter Defense Problem (PDP); Spiking Neural Network (SNN); Inhibitory connections; Greedy.

US

1. Introduction

The Perimeter Defense Problem (PDP) is a multiplayer problem, in which a team of intruders tries to enter a given (for ease assumed to be convex) territory, and a team of defenders (operating on the perimeter) tries to capture the intruders on the perimeter before the intruders enter the territory. The PDP was first proposed in [1] as a perimeter patrolling problem. Later, the PDP was addressed using geometrical approaches for a circular territory [2, 3], conical territory [4–6], convex territory [7–11] and recently for a 3D hemisphere territory [12]. A detailed review of different solution approaches and the challenges in solving a PDP was presented in [13].

An adaptive partitioning of space into disjoint regions to protect a circular territory was presented in [3]. The defenders were assigned to the disjointly partitioned regions in [3] to protect the perimeter from nonuniformly distributed intruders. A decentralized strategy for defending a convex-shaped territory was proposed in [9]. The solution in [9] solves a spatiotemporal multitask assignment problem in a decentralized manner for multirobot

Received 1 September 2024; Accepted 30 December 2024; Published 21 February 2025. This paper was recommended for publication in its revised form by editorial board member, Jianan Wang.
Email Addresses: ‡*pagalat@iisc.ac.in*, §*vssuresh@iisc.ac.in*, ¶*ensundara@gmail.com*, ‖*shrvel@ltu.se*
‡Corresponding author.

systems (DMRSTMTA). A multitask assignment approach for protecting a convex territory using a dynamic defender team size was proposed in [10]. The above solutions for solving the PDP require retraining when the shape of the perimeter or the size of the defender team changes. In [11], a Spiking Neural Network (SNN)-based solution for solving the PDP without any additional retraining when the defender team or perimeter shape changes was presented. Here, the SNNs were considered because they efficiently handle both space and time interrelations and can process information in an energy-efficient manner as events [14–16]. SNNs with different types of connections, such as inhibitory [17] and time-varying connections [18], have been explored in the literature. A detailed overview of SNN learning algorithms can be found in [19]. The solution proposed in [11] uses a Decentralized Spike-based Learning approach (DSL) for defending a convex-shaped territory. In DSL, the SNN was trained in a supervised manner with expert assignments obtained by a centralized ground-truth policy developed earlier in [10].

Another supervised learning-based solution for defending a 3D hemispherical territory using graph neural networks was presented in [12]. The solution proposed in [12] assigns one intruder to each defender; hence, it is not suitable when intruders are in higher numbers than those of the defenders. The maximum matching algorithm proposed in [20] was used as an expert policy to train a graph neural network in [12]. All the learning-based solutions proposed in [11, 12] require an external expert algorithm to obtain the desired assignments for training. In addition, because of the sparsity and imbalance in the data generated during training, DSL uses an additional oversampling technique, as in [21]. The use of additional algorithms to balance data and training may lead to an increase in computational complexity and additional memory requirements. Hence, there is a critical need for a learning-based solution for PDP that can generate expert data on its own for training without the need of any additional algorithms and this is exactly the problem that this study attempts to address.

In this study, a Centralized Conflict-free Assignment Learning Spiking neural network (C2ALS) is formulated for solving PDP. As the PDP can be formulated as a spatio-temporal problem (as discussed in [9]), SNNs are better tools for handling spatial and temporal data as events known as spikes [22, 23]. The SNN in the proposed approach is trained to learn the greedy assignments that can be generated with fewer computations, in contrast to that in [11], where the training data are generated by solving optimization problems. By greedy assignments to the defender, we refer to those assignments to a defender where the intruder locations on the perimeter that are closest to a defender are selected so that it can reach location I at the same time when an intruder arrives. These assignments are referred to as greedy assignments, as the defender is assigned to those locations that are closest to it compared to any farther locations. The capture assignments for each defender to protect the territory are obtained using C2ALS. Sometimes, these assignments may lead to conflicts, and to obtain assignments without conflicts, a conflict-free assignment learning SNN is used for learning the greedy assignment. If any conflict occurs in the assignments of the defenders, then with the help of inhibitory connections [17] these conflicts are resolved during the training of the SNN. In C2ALS, the updation strategy is designed to update the weights of these inhibitory connections along with other weights in the network. Unlike in [24], C2ALS does not implement an additional conflict resolution mechanism beyond the bidding algorithm used for multitask assignment.

The C2ALS is formulated by considering that the size of the team of defenders is smaller than that of the intruders at any given time. Hence, this necessitates at least one defender to perform multiple tasks of capturing the intruders to keep the perimeter safe. The performance of C2ALS is evaluated using the metric of success rates, which is defined by the percentage captures of intruders by a team of defenders to protect the territory. C2ALS performance is also compared with other existing one-to-many assignment-based solutions that are both centralized and decentralized. The results indicate that the proposed C2ALS performs 17% better than other existing state-of-the-art centralized one-to-many solutions, namely, Adaptive Partitioning (presented in [3]). Also, the proposed C2ALS approach outperforms DSL by 3% which is a decentralized learning-based one-to-many solution. A qualitative performance comparison demonstrates that the proposed C2ALS approach eliminates the need for additional conflict resolution or extensive expert algorithms (as the desired assignments are obtained using a simple greedy approach). Furthermore, the trained SNN in C2ALS can be directly utilized to determine assignments when the number of intruders or the perimeter shape changes, provided all other parameters remain constant. This paper is arranged as follows. Section 2 presents a detailed formulation of the spiking neural network for Centralized Conflict-free Assignment learning. The qualitative and quantitative performance evaluations of the proposed C2ALS approach are presented in Sec. 3. Finally, the conclusions of this study are summarized in Sec. 4.

2. Centralized Conflict-free Assignment Learning Spiking Neural Network (C2ALS)

In this section, the proposed centralized approach using a SNN to solve the PDP is discussed. First, an instantaneous

scenario of the perimeter defense problem with a higher number of intruders than the defenders is presented. Then the C2ALS formulation is presented.

2.1. *A typical scenario of the perimeter defense problem (PDP)*

The typical PDP scenario consists of intruders who try to enter through the perimeter of a protected territory, and defenders who protect the perimeter from these intrusions. Let us consider a total of D defenders are used to defend the perimeter of $N(> D)$ intruders. Figure 1 shows a typical PDP scenario, in which the region to be defended is shown in green. The boundary of the green region is considered to be the perimeter. The region between the perimeter and the sensor range for detection by the defender is divided into two different layers: a sensing layer and a capture layer. For ease of generalization, the shapes of the sensing and capture layers are considered similar to that of the perimeter. The region in which defenders can sense the intruder's information, such as the position and arrival time towards the perimeter, is termed the sensing layer. The region in which the defenders can capture the intruders is called the capture layer. The sensor layer is larger and encloses the capture layer. In the capture layer, a defender senses and also obtains information from other defenders. The defenders are constrained to operate only in the capture layer surrounding the perimeter, as shown in Fig. 1. Each layer is segmented into I angular segments. An ith angular segment s^i that makes an angle k° with the center of the region to be defended (i.e. the green region) is shown in Fig. 1. Intruders are assumed to be radially incoming towards the perimeter through these segments. A strategy needs to be developed to spatially intercept the intruders in the capture layer at their time of arrival. Because of the smaller number of defenders than the intruders, each defender must perform multiple spatiotemporal tasks to capture the intruders. The proposed centralized greedy assignment SNN for the PDP and its learning algorithm is presented in the following section.

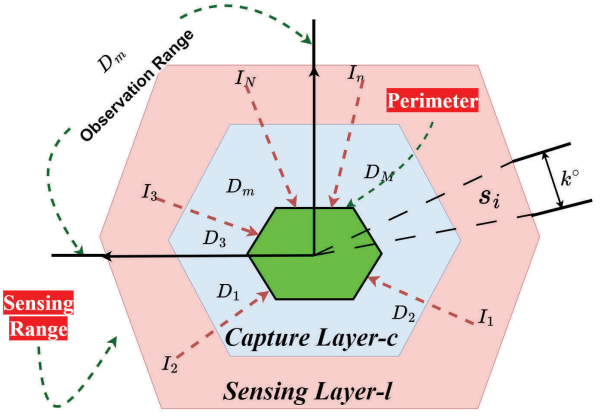


Fig. 1. A typical PDP scenario.

2.2. *Centralized conflict-free assignment learning spiking neural network (C2ALS) formulation*

A C2ALS to solve the PDP is presented in this section, and a schematic of the same is shown in Fig. 2. An SNN is trained to obtain conflict-free assignments using inhibitory connections among neurons in the distributed coding layer. The SNN in C2ALS is trained in a supervised manner to output the desired assignments, which are greedily obtained. These are termed greedy assignments because each defender is assigned to its closest intruder in the PDP scenario. Let us consider that an intruder radially arrives at segment s_j of the perimeter at time t_j. Let defender d be required to capture an intruder in segment s_k (which is the segment closest to s_j where a defender is already assigned) at time t_k s $(t_k < t_j)$. Depending upon the velocity of the defender d (V_d), its desired greedy assignment to segment s_j is

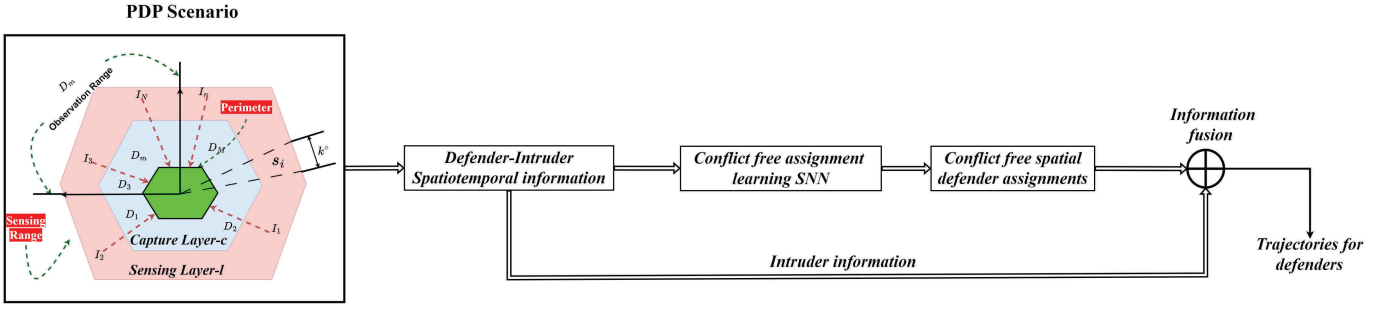


Fig. 2. Schematic representation of proposed centralized framework for conflict-free greedy assignment learning SNN-based solution for PDP.

evaluated as

$$\hat{y}_j^d = \begin{cases} 1, & \text{if } V_d \geq \dfrac{d_{kj}}{t_j - t_k}, \\ 0, & \text{else.} \end{cases} \tag{1}$$

The SNN developed to learn the aforementioned greedy assignments without any conflicts is presented in the following section.

2.2.1. *Conflict-free assignment learning SNN*

The perimeter is divided into 36 angular segments, each of which makes an angle of 10° with respect to the center. Each input neuron in the SNN is associated with one of the segments in the perimeter. Let us consider that an input neuron x_j is associated with the segment s_j in the perimeter. If a defender is already present in segment s_j, then the input neuron x_j provides a spike at the start of the simulation at 1 ms out of the total simulation interval T. Similarly, if an intruder I is sensed to radially incoming toward the segment s_j of the perimeter, then the input neuron x_j provides a spike at t_j^I as given as follows:

$$t_j^I = \frac{d_{Ij}^{\perp}}{V_{Ij}}, \tag{2}$$

where $d_{Ij}^{\perp}$ is the perpendicular distance between the sensed location of the intruder I and segment s_j on the perimeter. V_{Ij} is the velocity of intruder I in direction $d_I^{\perp}j$. The spike pattern generated by input neuron x_j is denoted by $t_j^x = \{t_j^{x_1}, \ldots, t_j^{x_f}\}$. Where $t_j^{x_f}$ is the fth input spike time generated by x_j neuron. In the Distributed Coding Assignment layer (DCA layer), two neurons $(a_j^d, \bar{a}_j^d)$ are used to decide the assignment of defender d to segment s_j. The neurons in the DCA layer are connected to the input neurons with the help of connections with weights w^1 as shown in Fig. 3. w_{jj}^{1d} and $w_{j\bar{j}}^{1d}$ represents the connections of the weights between input neuron x_j and DCA layer neurons a_j^d and $\bar{a}_j^d$ in the defender-d assignment part of the SNN. In the DCA layer, positive assignment neurons (a_j) among the different defenders are connected using inhibitory connections. h_{jj}^{1d} represents the weight of the inhibitory connection between a_j^1 neuron of the defender 1 assignment part of the SNN and a_j^d neuron of the defender d assignment part of the SNN. Each positive assignment neuron is inhibitory and connected only to its counterpart in the other defender assignment parts. The potential developed by the positive-assignment neuron a_j^d at a given time instant t in the DCA layer of the defender d assignment part of the SNN is denoted by $v_j^d(t)$.

$$v_j^d(t) = \sum_{i=1}^{36} w_{ij}^{1d} \cdot \sum_f \epsilon(t - t_i^{x_f}) - \sum_{k,k\neq d} h_{jj}^{kd} \epsilon(t - t_j^{a_k}), \tag{3}$$

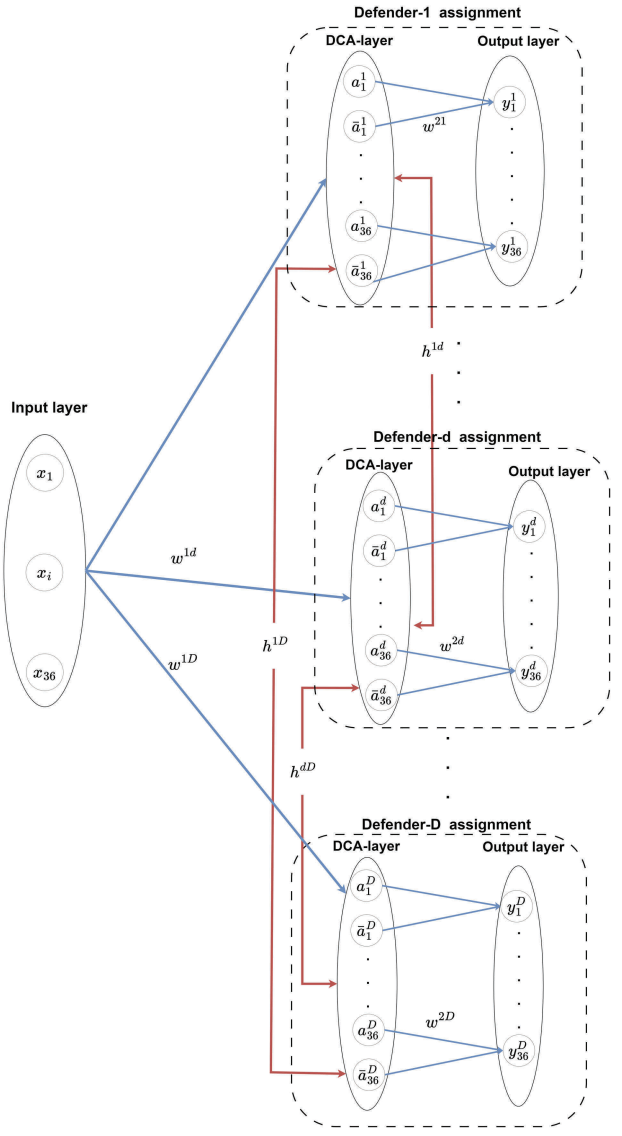


Fig. 3. Conflict-free assignment learning SNN.

where $t_j^{a_k}$ denotes the spike time of a_j^k neuron in DCA layer of defender-k assignment part of SNN. Similarly, the potential developed negative assignment neuron $\bar{a}_j^d$ is denoted by $\bar{v}_j^d(t)$.

$$\bar{v}_j^d(t) = \sum_{i=1}^{36} w_{i\bar{j}}^{1d} \cdot \sum_f \epsilon(t - t_i^{x_f}) \tag{4}$$

ϵ is the spike response function evaluated using [25]. If τ is the constant of the neuron, then ϵ is evaluated as

$$\epsilon(t) = \frac{t}{\tau} \cdot \exp\left(1 - \frac{t}{\tau}\right). \tag{5}$$

Based on the spike time $t_j^{a_d}$ of positive assignment a_j^d neuron and $t_{\bar{j}}^{a_d}$ of negative assignment $\bar{a}_j^d$ neuron the final assignment of defender d to segment s_j is obtained using y_j^d

output neuron.

$$y_j^d = \begin{cases} 1, & \text{if } t_j^{a_d} < t_{\bar{j}}^{a_d}, \\ 0, & \text{else if } t_j^{a_d} \geq t_{\bar{j}}^{a_d}. \end{cases} \tag{6}$$

If the output of y_j^d neuron is 1, then the defender d is assigned to segment s_j in the perimeter and vice versa if the output of y_j^d neuron is 0. The weights of the connections in the SNN are updated such that the output of y_j^d neurons matches its desired greedy assignment $\hat{y}_j^d$.

2.3. *Conflict-free assignment learning algorithm for SNN*

Depending on the desired and actual outputs of the neurons, the weights in the SNN are updated using a conflict-free assignment learning algorithm. The weights are updated according to any one or more of the following scenarios: skipped assignment, false assignment, or assignments with conflicts. A detailed description of these scenarios and the corresponding weight update equations for these scenarios are presented as follows:

- Skipped assignment (**If** $\hat{y}_j^d = 1$ and $\hat{y}_j^d \neq y_j^d$):
  In this scenario, the SNN does not assign defender-d to segment s_j while it is required to be assigned. Therefore, in this scenario, the SNN skips an assignment for defender d to segment s_j. From Eq. (6), it can be observed that the defender-d is not assigned to the segment s_j because the spike time of a_j^d neuron ($t_j^{a_d}$) is not lower than spike time of $\bar{a}_j^d$ neuron ($t_{\bar{j}}^{a_d}$). Therefore to make the actual output y_j^d equal to $\hat{y}_j^d$, the a_j^d neuron is made to spike earlier than the $\bar{a}_j^d$ neuron. This can be achieved by increasing the feed-forward weights connected to a_j^d and decreasing the weights connected to $\bar{a}_j^d$ neuron. Equations (7) and (8) show the weight update connected to a_j^d and $\bar{a}_j^d$ neurons using Synaptic Time-Dependent Plasticity (STDP) as in [26].

$$w_{ij}^{1d} \leftarrow w_{ij}^{1d} + \sum_{f, t_i^{x_f} < t_j^{a_d}} \eta^+ \cdot \exp\left(-\frac{t_j^{a_d} - t_i^{x_f}}{\tau_u}\right), \quad \forall i \in [1, 36] \tag{7}$$

$$w_{i\bar{j}}^{1d} \leftarrow w_{i\bar{j}}^{1d} - \sum_{f, t_i^{x_f} < t_j^{a_d}} \eta^- \cdot \exp\left(-\frac{t_{\bar{j}}^{a_d} - t_i^{x_f}}{\tau_u}\right), \quad \forall\, i \in [1, 36], \tag{8}$$

  where $\tau_u, \eta^+ \& \eta^-$ are the STDP time constant, learning rates during positive and negative weight updates, respectively.
- False assignment (**If** $\hat{y}_j^d = 0$ **and** $\hat{y}_j^d \neq y_j^d$):
  In this scenario, defender-d is falsely assigned to segment s_j, whereas the desired greedy assignment suggests that it should not be assigned. This occurs when y_j^d is equal to 1, which is because the spike time of the a_j^d neuron ($t_j^{a_d}$) is lower than the spike time of $\bar{a}_j^d$ neuron ($t_{\bar{j}}^{a_d}$) from Eq. (6). To make the actual assignment y_j^d identical to the desired assignment $\hat{y}_j^d$, the weights connected to a_j^d neuron are decreased and the weights connected to $\bar{a}_j^d$ neuron are increased such that the latter spikes earlier than the former. The required updates connected to a_j^d and $\bar{a}_j^d$ neurons in this scenario are as follows:

$$w_{ij}^{1d} \leftarrow w_{ij}^{1d} - \sum_{f, t_i^{x_f} < t_j^{a_d}} \eta^- \cdot \exp\left(-\frac{t_j^{a_d} - t_i^{x_f}}{\tau_u}\right), \quad \forall\, i \in [1, 36], \tag{9}$$

$$w_{i\bar{j}}^{1d} \leftarrow w_{i\bar{j}}^{1d} + \sum_{f, t_i^{x_f} < t_{\bar{j}}^{a_d}} \eta^+ \cdot \exp\left(-\frac{t_{\bar{j}}^{a_d} - t_i^{x_f}}{\tau_u}\right), \quad \forall\, i \in [1, 36]. \tag{10}$$

- Assignment with conflicts (**If** $\hat{y}_j^d = y_j^d = 1$ **and if** $\forall_{k,k\neq d}\ y_j^k = y_j^d$):
  Even though the actual assignment y_j^d is the same as the desired assignment $\hat{y}_j^d$, there is a conflict in the assignment in this scenario. This conflict arises because any other defender k is also assigned to segment s_j in the perimeter, along with defender d. The segment s_j should be assigned to defender d rather than defender k as the desired assignment $\hat{y}_j^d = 1$. Therefore, in this scenario, segment s_j should not be assigned to the defender k. This is achieved if a_j^k neuron spikes after the $\bar{a}_j^k$ neuron. This spiking phenomenon is achieved by increasing the weight of the inhibitory connections and decreasing the weight of the feedforward connections connected to a_j^k neuron. Figure 4 shows the resolution of conflicts using inhibitory connections.

$$w_{ij}^{1k} \leftarrow w_{ij}^{1k} - \sum_{f, t_i^{x_f} < t_j^{a_k}} \eta^- \cdot \exp\left(-\frac{t_j^{a_k} - t_i^{x_f}}{\tau_u}\right), \quad \forall i \in [1, 36] \tag{11}$$

$$h_{jj}^{ik} \leftarrow h_{jj}^{ik} + \eta^+ \cdot \exp\left(-\frac{t_j^{a_k} - t_j^{a_i}}{\tau_u}\right), \tag{12}$$

  where $t_j^{a_i} < t_j^{a_k}, \quad \forall\, i \in [1, D] \wedge i \neq k$.

Once the conflict-free assignments of all defenders are obtained using the SNN, they are combined with intruder information to obtain the final trajectories. Based on the time of arrival of the intruders the assignments of each defender are arranged sequentially while obtaining their

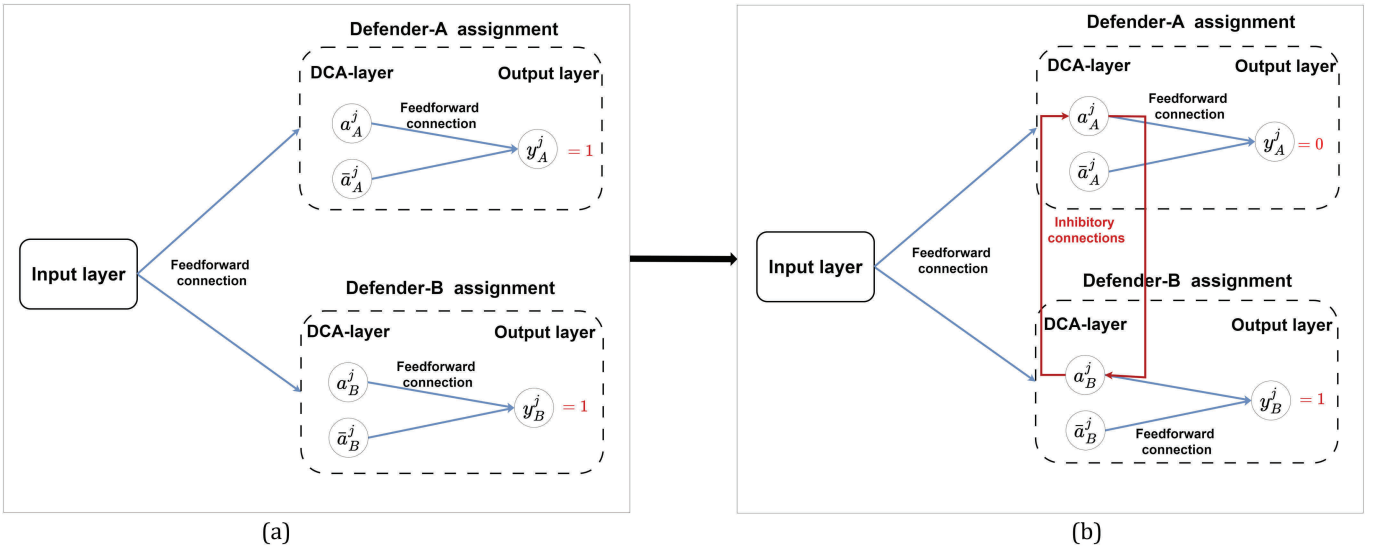


Fig. 4. (A) Conflict is detected between the assignments of Defenders *A* and *B*. (B) Conflict is resolved with the help of inhibitory connections in the DCA layer.

final trajectories. Each defender follows its trajectory in the capture layer and intercepts the intruders to defend the perimeter. The pseudocode for training the SNN to obtain desired and conflict-free assignments is presented in Algorithm 1.

3. Performance Evaluation of C2ALS

In this section, the detailed performance evaluation of the C2ALS proposed for solving a PDP is presented. Initially, the data generation method for training the C2ALS is discussed along with the simulation setup. Then a case study presenting the capture assignments obtained using the C2ALS for a given PDP scenario is presented. Then, the overall success percentage of the proposed C2ALS in defending the territory is presented. To study the effectiveness of the C2ALS, a comparison with other state-of-the-art centralized and decentralized methods is also presented.

Algorithm 1. Pseudocode for training Conflict-free assignment SNN

for each training sample obtain the desired and final assignments of all output neurons using Eqs. (1) and (6) **do**
for each d ∈ Total number of defenders (D) **do**
for each j ∈ Total number of segments (36) **do**
if $\hat{y}_j^d = 1$ and $\hat{y}_j^d \neq y_j^d$ **then**
update weights in DCA layer using Eqs. (7) and (8)
end if
if $\hat{y}_j^d = 0$ and $\hat{y}_j^d \neq y_j^d$ **then**
update weights in DCA layer using Eqs. (9) and (10)
end if
if $\hat{y}_j^d = y_j^d = 1$ and $\forall_{k,k\neq d}\ y_j^k = y_j^d$ **then**
update weights in DCA layer using Eqs. (11) and (12)
end if
end for
end for
end for

3.1. *Simulation setup*

In this study, a PDP problem is considered, where the intruders arrive with a Poisson distribution with an arrival rate of 4 over 8 s. A segment with $k = 10^\circ$ resulting in 36 segments is considered. These intruders are placed randomly across the segments in the sensing layers. A team of $D = 5$ defenders is considered for training and to assign tasks among all defenders, each defender is restricted to capture a maximum of 3 intruders. This defender team is also placed randomly in the capture layer ensuring not more than one defender is present in a segment. The time constant of the spiking neuron τ in the DCA layer is set to 0.5 s whereas the time constant for update τ_u is set to 0.3 s in the C2ALS. The positive (η^+) and negative learning rates (η^-) in the C2ALS are set to 1e-3 and 0.5e-3, respectively.

A total of 1200 samples of PDP scenes are generated for training the SNN in C2ALS. Out of these 1200 samples, 1000 samples are considered for training and the remaining 200 for testing. Each sample represents a PDP scenario that

includes the spatiotemporal information of the defenders and intruders along with their desired assignments. The number of features in each sample is equal to the number of segments. Each feature in a sample represents the spatiotemporal data of the defender or the intruder in a segment. The desired assignments are obtained using a greedy approach.

The simulations were carried out in a Windows 10 system using a MATLAB 2023a environment on a 3.2 GHz machine with 6 cores and 16 GB memory. The success rate is used as a metric to evaluate the performance of the different existing solutions for PDP (including the proposed solution using C2ALS). The percentage of intruders collectively captured by the team of defenders is termed the success rate.

$$\text{Success rate} = \frac{\text{Number of captured intruders}}{\text{Total number of intruders}} \times 100. \tag{13}$$

3.2. *Case study*

Let us assume a PDP scenario with 2 defenders D_p and D_q, where D_p is located at the segment s_p and D_q is located at the segment s_q at the start of the simulation, i.e. at t_0, respectively, as shown in Fig. 5. Let V_p and V_q be the velocities of the defenders D_p, & D_q, respectively, and each defender can capture a maximum of 2 intruders. Similarly, intruders

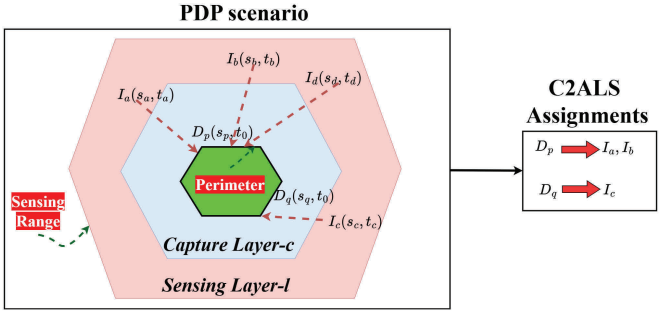


Fig. 5. PDP scenario case study.

I_a, I_b, I_c and I_d are trying to enter the parameter radially from segments s_a, s_b, s_c and s_d and their arrival times are t_a, t_b, t_c and t_d, respectively. Let the intruder I_c arrive much later than intruder the I_b (i.e. $t_c \gg t_b$), and the intruder I_d arrives immediately after intruder I_c (i.e. $t_d - t_c \approx 0$). Based on Eq. (1) the C2ALS is trained to assign intruders I_a and I_b for the defender D_p and the intruder I_d for the defender D_q, respectively, as shown in Fig. 5. Defender D_p cannot be assigned to intruder I_d based on Eq. (1) as any defender can capture only a maximum of 2 intruders. Defender D_q also cannot be assigned to capture I_d because it is present in segment s_c at time t_c before the arrival of intruder I_d (as it is assigned to capture intruder I_d) and $V_q < \frac{|s_d - s_c|}{|t_d - t_c|}$ as $t_d - t_c \approx 0$. The qualitative performance analysis of the proposed C2ALS approach is presented in the following section.

3.3. *Qualitative comparison of C2ALS with other existing approaches*

Table 1 shows a qualitative comparison of the proposed C2ALS with other existing state-of-the-art (SOTA) solutions. Other existing SOTA solutions used for comparison are Adaptive Partitioning, DMRSTMTA, Modified DMRSTMTA, and DSL. Adaptive Partitioning (AP) is an optimization-based one-to-many assignment-based solution for PDP. The AP solution for the PDP has two stages: partitioning and assignment. Modified DMRSTMTA is a modified version of DMRSTMTA [9] in which a distance-based cost function is used to obtain the optimal assignments for the defender. DSL is a learning-based approach for solving PDP that uses Modified DMRSTMTA assignments as an expert for training. [11]. The C2ALS approach is a one-to-many learning-based assignment solution for PDP. From Table 1, it can be observed that the proposed C2ALS requires expert greedy assignments with fewer computations for training. Similarly, due to the inhibitory connections, the C2ALS approach outputs conflict-free defender assignments and hence does not require any additional approach to resolve conflicts. However, DMRSTMTA and DSL require additional conflict-resolving algorithms owing to decentralized training.

Table 1. Qualitative comparison of C2ALS with other existing state-of-the-art solutions for PDP.

Algorithm	Expert for Training	Conflict-free resolving algorithm	Defender team changes	Perimeter shape changes
AP	NA	NA	Needs Re-solving	Needs Re-solving
DMRSTMTA	NA	Required	Needs Re-solving	Needs Re-solving
Modified DMRSTMTA	NA	Required	Needs Re-solving	Needs Re-solving
DSL	Required	Required	No retraining	No retraining
C2ALS	Not required	Not required	Needs Retraining	No retraining

The time complexity of the conflict-free resolving algorithm used in both DMRSTMTA and DSL is of the order $O(n)$, where n is the total number of assignments for a defender. The proposed C2ALS has lesser time complexity than DMRSTMTA and DSL because it generates conflict-free assignments using inhibitory connections in the architecture without the use of an additional algorithm. Next, a quantitative comparison of C2ALS is presented.

3.4. *Quantitative comparison of the C2ALS with other existing approaches*

Table 2 shows a quantitative comparison of the proposed C2ALS for solving PDP with other existing SOTA solutions. The success rate in Table 2 is the average success percentage of each approach in protecting the perimeter for over 1000 trials. The performance of C2ALS is 17% higher than that of the adaptive partitioning approach. Compared with existing SOTA decentralized approaches, C2ALS has a 3% higher success rate. The assignments obtained in C2ALS are conflict-free, whereas in decentralized approaches, there is a high probability of conflict in the assignments. Due to this, decentralized approaches like DMRSTMTA and DSL use an additional conflict-free resolving algorithm which is not required in C2ALS. In addition, the C2ALS approach is easily generalizable compared to other approaches to handle changes in the perimeter's shape and the intruders' velocities.

3.5. *Discussion*

3.5.1. *Effect on the performance of C2ALS due to change in velocity ratio*

The performance of the proposed C2ALS for PDP is affected by both the velocity of the defender and the incoming intruder. Figure 6 shows the variation in the average success rate (over 1000 runs) of the C2ALS approach in defending the territory when the ratio of the defender velocity to an intruder velocity is changed. V_D is the velocity of the defender parallel to the perimeter. V_I is the incoming velocity of the intruder perpendicular to the perimeter. From Fig. 6, it can be observed that the performance of the C2ALS in defending the perimeter increases when V_D is greater than V_I.

Table 2. Quantitative comparison of C2ALS with other existing SOTA solutions for PDP.

Algorithm	Method	Success rate (Mean ± SD)
Centralized algorithms		
Adaptive Partitioning	Optimization based	64.90 ± 3.2
C2ALS	Learning based	82.85 ± 2.6
Decentralized algorithms		
DMRSTMTA	Optimization based	52.19 ± 8.9
Modified DMRSTMTA	Optimization based	68.23 ± 13.4
DSL	Learning based	79.32 ± 10.8

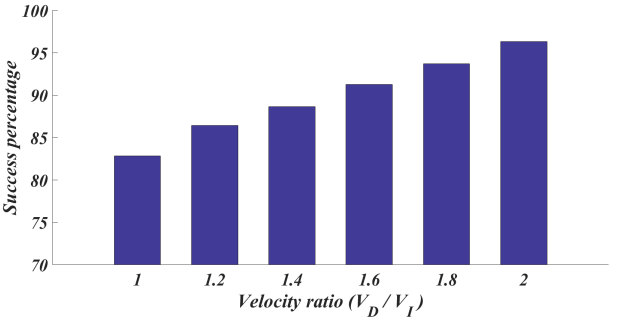


Fig. 6. Variation in success rates of C2ALS to change in velocity ratio.

3.5.2. *Effects due to change in the number of intruders*

Figure 7 illustrates the variation in the C2ALS success rate, the average number of defenders assigned, and the average segments traveled by a defender as the number of intruders in a given PDP scenario is increased. These experiments are carried out by varying the velocity ratio ($\frac{V_D}{V_I}$) from 1 to 2 in steps of 0.2. A dip in success rates, as shown in Fig. 7(a), occurs as the number of intruders increases, which can be attributed to the limited number of defenders held constant throughout these experiments From Fig. 7(b) it can be observed that the average number of intruders assigned to each defender is between 1 and 3 across all velocity ratios. The number of tasks assigned by the proposed C2ALS approach to each defender increases with an increase in the number of intruders. The average number of segments a defender travels from its initial position to capture intruders remains relatively constant, around 8–11, even as the number of intruders and velocity ratios vary, as shown in Fig. 7(c). Consequently, C2ALS achieves a success rate of 65–98%, with each defender covering approximately 30.56% (11 out of 36 segments) of the total perimeter length when the number of intruders is nearly doubled. This study suggests that C2ALS minimizes unnecessary movement, potentially leading to lower operational costs and quicker response times.

3.5.3. *Effect on performance of C2ALS due to change in the number of defenders*

The number of neurons in the DCA and output layer of SNN used in C2ALS is proportional to the number of defenders in a given PDP scenario as in Fig. 3. Therefore, a change in the

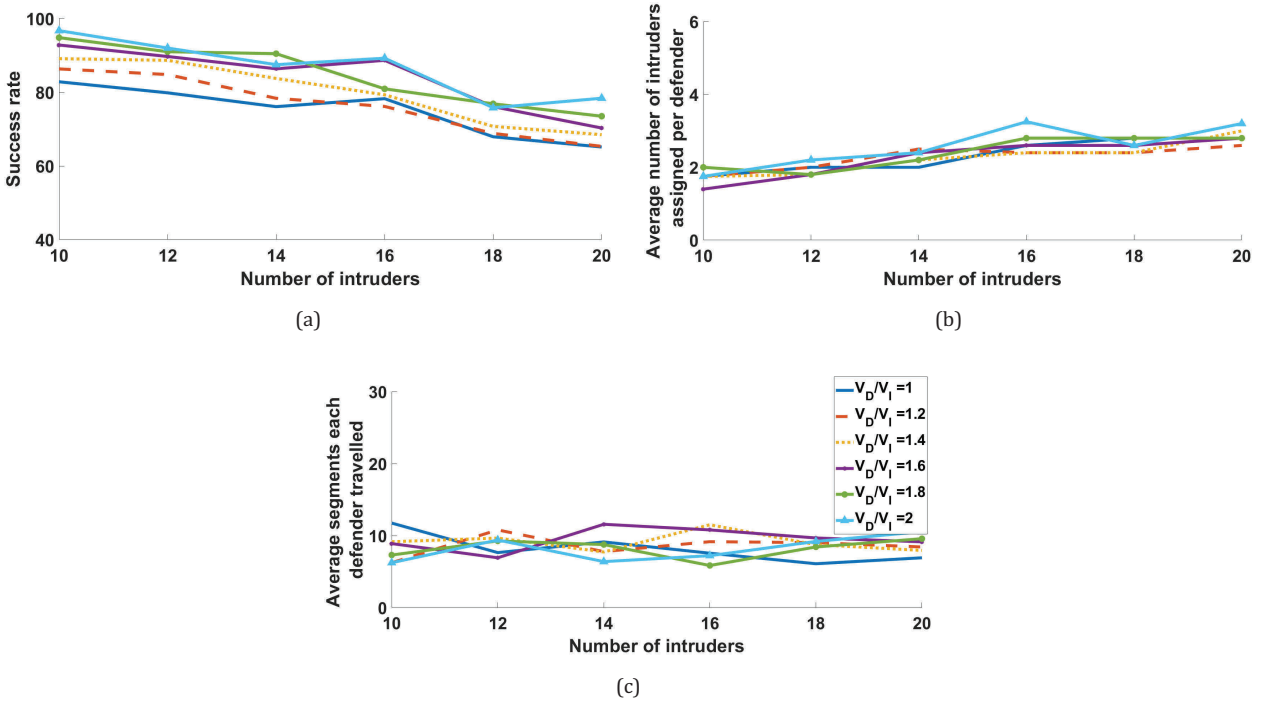


Fig. 7. Variation in (a) Success rate, (b) Average number of intruders assigned per defender and (c) Average number segments traveled by a defender, due to variation in number of intruders in a given PDP scenario for different velocity ratios (i.e. $\frac{V_D}{V_I} = 1, 1.2, 1.4, 1.8$ and 2, respectively.)

number of defenders affects the performance of the proposed C2ALS approach. However, if the number of defenders is increased in maintaining the constant number of intruders, then the success rate also increases as more defenders are available to defend the perimeter.

3.5.4. *Effect on performance of C2ALS due to change in the size/shape of the perimeter*

The change in the perimeter's size and shape has an impact on the C2ALS approach's performance. If the size/shape of the perimeter is changed, then there will be changes in the arrival times of the intruders. Further, this will affect the input spike patterns generated by each input neuron in the SNN. However, the number of neurons in the SNN remains same and it need not be altered if the perimeter's size is maintained constant when changing its shape. In this case, the earlier proposed SNN model can be used to evaluate the success rate without any additional retraining. However, if the size of the perimeter is also changed, the number of neurons required also changes which further necessitates additional training of the SNN.

3.5.5. *Computations required for training the SNN using C2ALS*

During the training of the SNN in the C2ALS approach, three different weight updates are carried out as mentioned in Sec. 2.3 namely: a skipped assignment, a false assignment, and an assignment with conflicts. Based on the existing conditions, any one of the Skipped and false assignment strategies may be required during training. Skipped and false assignment strategies perform two weight updates connected to a single input neuron for each segment to obtain assignments for a single defender. Therefore for D number of defenders and N_s number of segments the computations required to perform each of these strategies is $2 * {N_s}^2 * D$. Assignment with conflicts weight update strategy also performs two weight updates as mentioned in Sec. 2.3 for each segment and for each defender. Therefore, assignments with conflicts weight update strategy also require $2 * {N_s}^2 * D$ computations, for D number of defenders and N_s number of segments. Thus during the training of SNN using the C2ALS approach the total number of computations required for weight update are of the order $O(4 * {N_s}^2 * D)$.

4. Conclusions

In this paper, a centralized greedy assignment learning algorithm developed for solving a PDP using an SNN is presented. Intruders are sensed in the sensing layer, and defenders are constrained to operate in the capture layer of the region. The angular segments in the capture layer to

which a defender is assigned to travel while protecting the perimeter are obtained using the SNN. The expert assignment for training the SNN is obtained based on the greedy assignments (nearest intruder assignment) generated using the feasibility of the solution. A C2ALS approach for solving a PDP is presented in this paper. Due to the inhibitory connections in the SNN, the C2ALS approach inherently generates conflict-free assignments for defenders without the use of an external conflict-free trajectory generation algorithm. The C2ALS approach successfully captures 82.85% of intruders that try to attack the perimeter, which is 17% higher than that of other existing one-to-many centralized adaptive partitioning approaches. In addition, the proposed approach has a 3% higher success rate than other SOTA decentralized approaches. Experimental results validate the robustness of C2ALS, demonstrating its ability to maintain efficient performance with defenders traversing only 8–11 of 36 perimeter segments, even as the number of intruders varies. Owing to expert greedy assignments, the proposed approach results in fewer computations compared to other existing learning-based solutions for PDP.

ORCID

Mohammed Thousif https://orcid.org/0000-0002-9292-4408
Shridhar Velhal https://orcid.org/0000-0002-6906-653X
Suresh Sundaram https://orcid.org/0000-0001-6275-0921
Narasimhan Sundararajan https://orcid.org/0000-0002-6972-8775

References

[1] N. Agmon, S. Kraus and G. A. Kaminka, Multi-robot perimeter patrol in adversarial settings, in *2008 IEEE Int. Conf. Robotics and Automation* (IEEE, 2008), pp. 2339–2345.
[2] S. Bajaj and S. D. Bopardikar, Dynamic boundary guarding against radially incoming targets, in *2019 IEEE 58th Conf. Decision and Control (CDC)* (IEEE, 2019), pp. 4804–4809.
[3] D. G. Macharet, A. K. Chen, D. Shishika, G. J. Pappas and V. Kumar, Adaptive partitioning for coordinated multi-agent perimeter defense, in *2020 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)* (IEEE, 2020), pp. 7971–7977.
[4] S. Bajaj, E. Torng, S. D. Bopardikar, A. Von Moll, I. Weintraub, E. Garcia and D. W. Casbeer, Competitive perimeter defense of conical environments, *2022 IEEE 61st Conf. Decision and Control (CDC)* (IEEE, 2022), pp. 6586–6593.
[5] S. Bajaj, S. D. Bopardikar, A. Von Moll, E. Torng and D. W. Casbeer, Perimeter defense using a turret with finite range and startup time, *2023 American Control Conf. (ACC)* (IEEE, 2023), pp. 3350–3355, doi: 10.23919/ACC55779.2023.10155838.
[6] S. Bajaj, S. D. Bopardikar, E. Torng, A. Von Moll and D. W. Casbeer, Multivehicle perimeter defense in conical environments, *IEEE Trans. Robot.* **40** (2024) 1439–1456.
[7] D. Shishika and V. Kumar, Local-game decomposition for multiplayer perimeter-defense problem, *2018 IEEE Conf. Decision and Control (CDC)* (IEEE, 2018), pp. 2093–2100.
[8] D. Shishika, J. Paulos and V. Kumar, Cooperative team strategies for multi-player perimeter-defense games, *IEEE Robot. Autom. Lett.* **5**(2) (2020) 2738–2745.
[9] S. Velhal, S. Sundaram and N. Sundararajan, A decentralized multi-robot spatiotemporal multitask assignment approach for perimeter defense, *IEEE Trans. Robot.* **38**(5) (2022) 3085–3096.
[10] S. Velhal, S. Sundaram and N. Sundararajan, Dynamic resource allocation with decentralized multi-task assignment approach for perimeter defense problem, *IEEE Trans. Aerosp. Electron. Syst.* **58**(4) (2022) 3313–3325.
[11] M. Thousif, S. Velhal, S. Sundaram and S. Dora, A decentralized spike-based learning framework for sequential capture in discrete perimeter defense problem, arXiv:2305.16748.
[12] E. S. Lee, L. Zhou, A. Ribeiro and V. Kumar, Graph neural networks for decentralized multi-agent perimeter defense, *Front. Control Eng.* **4** (2023) 1.
[13] D. Shishika and V. Kumar, A review of multi agent perimeter defense games, *Int. Conf. Decision and Game Theory for Security* (Springer, 2020), pp. 472–485.
[14] W. Maass, Noisy spiking neurons with temporal coding have more computational power than sigmoidal neurons, *Advances in Neural Information Processing Systems 9*, eds. M. C. Mozer, M. I. Jordan and T. Petsche (MIT Press, 1997), pp. 211–217.
[15] D. Wu, G. Jin, H. Yu, X. Yi and X. Huang, Optimising event-driven spiking neural network with regularisation and cutoff, arXiv: 2301.09522.
[16] T. Tsuji, T. Orima and Y. Horio, An event-driven mixed analog/digital spiking neural network circuit model for hippocampal spatiotemporal context learning and memory, *2024 Int. Joint Conf. Neural Networks (IJCNN)* (IEEE, 2024), pp. 1–8, doi: 10.1109/IJCNN60899.2024.10650476.
[17] P. Machingal, Thousif, S. Dora, S. Sundaram and Q. Meng, Learning to classify faster using spiking neural networks, *2023 Int. Joint Conf. Neural Networks (IJCNN)* (IEEE, 2023), pp. 1–8, doi: 10.1109/IJCNN54540. 2023.10191334.
[18] A. Jeyasothy, S. Sundaram and N. Sundararajan, SEFRON: A new spiking neuron model with time-varying synaptic efficacy function for pattern classification, *IEEE Trans. Neural Netw. Learn. Syst.* **30**(4) (2018) 1231–1240.
[19] S. Dora and N. Kasabov, Spiking neural networks for computational intelligence: An overview, *Big Data Cognit. Comput.* **5**(4) (2021) 67.
[20] M. Chen, Z. Zhou and C. J. Tomlin, Multiplayer reach-avoid games via low dimensional solutions and maximum matching, *2014 American Control Conf.* (IEEE, 2014), pp. 1444–1449.
[21] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, Smote: Synthetic minority over-sampling technique, *J. Artif. Intell. Res.* **16** (2002) 321–357.
[22] L. Ning, J. Dong, R. Xiao, K. C. Tan and H. Tang, Event-driven spiking neural networks with spike-based learning, *Memetic Comput.* **15**(2) (2023) 205–217.
[23] P. Kang, S. Banerjee, H. Chopp, A. Katsaggelos and O. Cossairt, Event — driven tactile learning with location spiking neurons, *2022 Int. Joint Conf. Neural Networks (IJCNN)* (IEEE, 2022), pp. 1–9, doi: 10.1109/IJCNN55064.2022.9892074.
[24] Q. Liu, K. Huang, H. Liu, R. Wang and G. Cheng, Multi-domain collaborative task allocation and conflict resolution in unmanned systems under complex constraints, *Unmanned Syst.* **0**(0) (2024) 1–11.

[25] S. M. Bohte, J. N. Kok and H. La Poutr, Error-backpropagation in temporally encoded networks of spiking neurons, *Neurocomputing* **48** (2002) 17–37.

[26] H. Markram, J. Lübke, M. Frotscher and B. Sakmann, Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs, *Science* **275**(5297) (1997) 213–215.

Mohammed Thousif is working towards the Ph.D. degree with the Department of Aerospace Engineering, Indian Institute of Science, Bengaluru, India. His research focuses on spiking neural networks, learning algorithms, and multi-label classification tasks, with applications in neural optimization and perimeter defense problems.

Shridhar Velhal is currently a PostDoctoral Fellow in Department of Computer, Electrical and Space Engineering, Luleå University of Technology, 971 87 Luleå, Sweden. He has obtained Ph.D. degree from Department of Aerospace Engineering, Indian Institute of Science, Bengaluru, India. His research interests include dynamics and control of autonomous systems, with a particular focus on multiagent systems, territory protection games, spatiotemporal tasks, and multitask allocation.

Suresh Sundaram (Senior Member, IEEE) received the Ph.D. degree in aerospace engineering from the Indian Institute of Science, Bengaluru, India, in 2005. He is currently a Professor with the department of Aerospace Engineering, Indian Institute of Science. From 2010 to 2018, he was an Associate Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include flight control, unmanned aerial vehicle design, machine learning, optimization, and computer vision.

Narasimhan Sundararajan (Life Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of Illinois at Urbana–Champaign, Urbana, IL, USA, in 1971. From 1971 to 1991, he was with the Vikram Sarabhai Space Centre, Indian Space Research Organization, Trivandrum, India. Since 1991, he had been a Professor (Retd.) with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He is a Technical Consultant in WIRIN Project with the Indian Institute of Science, Bengaluru, India. His current research interests include spiking neural networks, neuro-fuzzy systems, and optimization with swarm intelligence.