Neuromorphic Digital-Twin-Based Controller for Indoor Multi-UAV Systems Deployment

Reza Ahmadvand, Sarah Safura Sharif, and Yaser Mike Banad

***Abstract*—This study introduces a novel distributed cloud-edge framework for autonomous multi-unmanned aerial vehicle (UAV) systems that combines the computational efficiency of neuromorphic computing with nature-inspired control strategies. The proposed architecture equips each UAV with an individual spiking neural network (SNN) that learns to reproduce optimal control signals generated by a cloud-based controller, enabling robust operation even during communication interruptions. By integrating spike coding with nature-inspired control principles inspired by tilapia fish territorial behavior, our system achieves sophisticated formation control and obstacle avoidance in complex urban environments. The distributed architecture leverages cloud computing for complex calculations while maintaining local autonomy through edge-based SNNs, significantly reducing energy consumption and computational overhead compared to traditional centralized approaches. Our framework addresses critical limitations of conventional methods, including the dependence on premodeled environments, computational intensity of traditional methods, and local minima issues in potential field approaches. Simulation results demonstrate the system's effectiveness across two different scenarios: first, the indoor deployment of a multi-UAV system made up of 15 UAVs, and second, the collision-free formation control of a moving UAV flock, including six UAVs considering the obstacle avoidance. Due to the sparsity of spiking patterns, and the event-based nature of SNNs on average for the whole group of UAVs, the framework achieves almost 90% reduction in computational burden compared to traditional von Neumann architectures implementing traditional artificial neural networks.**

***Index Terms*—Cloud-edge computing, distributed control, formation control, multi-UAV systems, nature-inspired control, spiking neural networks (SNNs).**

I. Introduction

THE evolution of autonomous robotic systems, particularly in multiagent scenarios, faces significant challenges in managing task complexity, optimizing energy consumption, and ensuring safe operation in complex environments. While swarm intelligence and nature-inspired approaches have demonstrated remarkable potential in decentralized control [1], the implementation of these systems often struggles with computational efficiency and real-time processing demands, particularly when scaling to larger swarms. Simultaneously, advances in neuromorphic computing systems [2] present promising solutions to these computational challenges, suggesting a potential synergy between nature-inspired control architectures and efficient computing paradigms for distributed multiagent systems. Traditional centralized control methods impose significant constraints on implementing sophisticated multiagent systems, particularly in urban environments where complex obstacles and dynamic conditions prevail [3], [4]. While cloud-based control architectures [5], [6] have emerged as potential solutions, they introduce additional complexities, including the requirement for persistent plant-cloud communication and potential vulnerabilities in case of communication failures. These limitations become particularly critical in multi-UAV systems, where reliable operation must be maintained even in cases of central computer failure [3], [4]. The need for robust decentralized control solutions that can maintain performance even with limited or interrupted cloud connectivity has become increasingly apparent. Spiking neural networks (SNNs) have emerged as a compelling solution for energy-efficient low-latency control applications [7], while nature-inspired control methods have demonstrated exceptional capability in handling complex multiagent scenarios [8], [9]. Traditional approaches to multi-UAV control face several critical limitations. Path planning and geometric guidance methods require a priori knowledge of the implementation environment with premodeled obstacles, limiting their effectiveness in dynamic or unknown environments [8], [9]. Potential field function approaches, while offering reactive control capabilities [10], can suffer from local minima problems and may not guarantee optimal trajectories. Model-predictive control methods [11], [12], though capable of anticipating future states, introduce significant computational overhead due to their requirement for continuous prediction of system dynamics over future time intervals. Despite recent advances combining standalone path planning with particle swarm optimization methods [13], [14], these hybrid approaches still struggle with real-time adaptation to dynamic obstacles [4]. Furthermore, while artificial potential field (APF) methods have shown promise in environments with known obstacle maps [15], [16], [17], [18], their effectiveness diminishes in scenarios with dynamic obstacles and multiple moving agents. These limitations become particularly apparent in urban

Received 3 February 2025; revised 26 March 2025; accepted 3 May 2025. Date of publication 6 May 2025; date of current version 23 May 2025. This work was supported in part by NASA Oklahoma EPSCoR Initialization under federal award no. 80NSSCM0029. *(Corresponding author: Yaser Mike Banad.)*

The authors are with the School of Electrical and Computer Engineering, University of Oklahoma, Norman, OK 73019 USA (e-mail: iamrezaahmadvand1@ou.edu; s.sh@ou.edu; bana@ou.edu).

Datasets are available online at https://github.com/INQUIRELAB/SNN-Based-UAV-Formation-Control

Digital Object Identifier 10.1109/JISPIN.2025.3567374

environments, where complex obstacles, including both static structures and dynamic obstructions, require real-time adaptation and decision-making capabilities. Inspired by digital-twin (DT) approaches, our research introduces a novel framework that synthesizes these parallel developments by implementing a distributed control architecture where each UAV is equipped with its SNN-based controller. This approach combines energy efficiency, low latency, and robustness of SNNs with nature-inspired control strategies. The system leverages the cloud in a neuromorphic DT architecture for complex calculations and control signal generation while maintaining local autonomy through individual SNNs that can learn and reproduce these control signals at the edge for each UAV separately. Building upon previous works in SNN-based frameworks for estimation and control of dynamical systems [19], [20] and recent advances in multi-UAV control systems incorporating APF concepts [15], [16], [17], [18], we propose a distributed cloud-edge architecture.

1) Each UAV is equipped with an individual SNN that learns to replicate control signals generated by a sophisticated controller operating on the cloud in the neuromorphic DT architecture.
2) The cloud controller implements a nature-inspired collision-free formation control strategy based on tilapia fish territorial behavior and obstacle avoidance [21], [22].
3) Each SNN operates independently from the implemented control algorithms in the cloud, ensuring the continued operation of the SNN-based controller at the edge even during discontinuous cloud-edge communication.
4) The distributed architecture maintains the benefits of swarm intelligence for the situational awareness of the agents while incorporating efficient neuromorphic computing at the individual agent level.

The foundation of our approach integrates multiple innovative elements from nature-inspired formation control drawing from biological models [4], individual SNNs implementing leaky integrate-and-fire (LIF) neurons [23], and obstacle avoidance strategies [24], [25], [26], implemented through neuromorphic computing. Our methodology uniquely combines rapid learning capabilities at the edge with nature-inspired control principles in the cloud. This distributed architecture ensures that each agent can maintain effective control even during temporary losses of cloud connectivity. When the connection is available, agents still benefit from the complex computational capabilities of cloud-based controllers. Furthermore, to assess the validity of our approach, it has been utilized in two different multi-UAV scenarios. First, the collision-free optimal indoor deployment of a multi-UAV system consists of 15 UAVs, and second, it has been applied to the collision-free formation control of a moving flock of six UAVs in the presence of obstacles. The results demonstrated that our strategy has an acceptable performance in the implementation of our controller architecture.

The rest of this article is organized as follows. Section II presents the theoretical framework, including the integration of distributed SNNs with nature-inspired control strategies, multi-UAV dynamic modeling, and the underlying principles of both cloud- and edge-based control. Section III details our implementation methodology, SNN architecture, and experimental setup. Section IV presents comprehensive simulation results and analysis across various scenarios. Section V concludes this article.

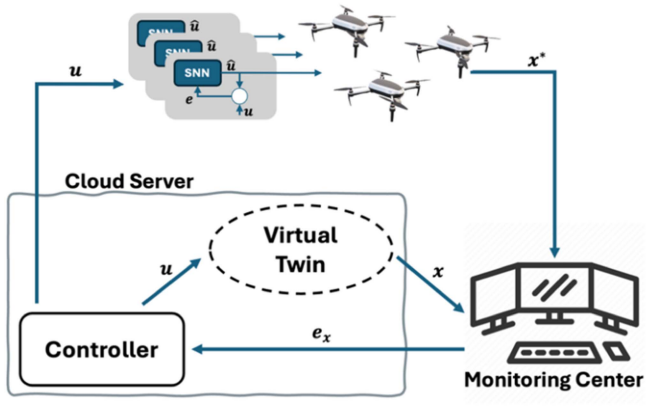

Fig. 1. Schematic of neuromorphic cloud-based DT.

II. Theory

This section begins with the theoretical underpinnings and outcomes of our study, focusing on the application of SNNs in our proposed architecture, and follows with controller design. Our approach leverages a nature-inspired controller on the cloud for collision-free optimal control of a multi-UAV system in a predefined barrier area [27]. The controller on the cloud interacts with a virtual twin network (a mathematical model on the cloud server, which models the physical multi-UAV system) to produce the ideal output of the real-world multi-UAV system as an expected output for comparison with the obtained outputs of the navigation system in the monitoring center. In the presented approach, each UAV has its own SNN-based controller that can learn to replicate the control signal $\boldsymbol{u}$ provided by the cloud using a local online supervised learning approach [26], [28], [29]. Fig. 1 depicts the proposed architecture for neuromorphic DT.

Moreover, our proposed architecture in this study leverages the advantages of neuromorphic computing such as the energy efficiency of SNNs, compared to traditional computing frameworks such as artificial neural networks (ANNs), while using the optimality of nature-inspired methods for environmental awareness.

A. SNN-Based Edge Controller

A spiking neural network (SNN)-based cloud-edge framework for a single-agent system was previously introduced in [26]. Building upon this foundation, the objective of this section is to extend that framework to support a multi-UAV system by deploying individual SNN-based controllers at the edge of each UAV. To set the stage for this extension, we begin by briefly reviewing the fundamental concepts of SNNs. SNNs are biologically inspired brain-mimicking computing tools that can efficiently replicate the neural circuits in the biological brain in which the neurons communicate via electrochemical signals called spikes. Based on the literature and engineering applications of SNNs, the focus of the presented study is on utilizing networks consisting of leaky LIF neurons [30].

Generally, to develop a network of LIF neurons capable of replicating a desired signal like $\boldsymbol{x}$, there are two strict assumptions that need to be satisfied. First, it is required to define a linear transformation like (1) for the extraction of the desired signals from the filtered spike trains $\boldsymbol{r}$

$$\hat{\boldsymbol{x}} = D\boldsymbol{r} \tag{1}$$

In the aforementioned equation, $\hat{}$ means that the parameter is extracted from neural activity, and D is a fixed random decoding matrix that can extract the desired signal $\hat{\boldsymbol{x}}$ from the filtered spike train vector $\boldsymbol{r}\in R^N$ [23], [30], [31], [32]. Filtered spike trains have their own dynamics, which obey the following expression::

$$\dot{\boldsymbol{r}} = -\lambda \boldsymbol{r} + \boldsymbol{s}. \tag{2}$$

In other words, filtered spike trains are the convolution of spike trains with synaptic kernels and can be interpreted as instantaneous firing rates, where $\boldsymbol{s}\in R^N$ is emitted spike train by the neurons in each time step and has a faster dynamic than $\boldsymbol{r}$ [23]. Thus, by optimizing the timing of spike occurrence instead of changing the output kernel values D, the SNN works in a way to minimize the cumulative prediction error, which is the error between the actual value of $\boldsymbol{x}$ and the extracted value from neural activity $\hat{\boldsymbol{x}}$. To achieve this, it minimizes the following cost function [2], [5]:

$$J = \frac{1}{t}\int_0^{\mathrm{t}} \left(\|\boldsymbol{x}(\tau) - \hat{\boldsymbol{x}}(\tau)\|_2^2 + \nu\|\boldsymbol{r}(\tau)\|_1 + \mu\|\boldsymbol{r}(\tau)\|_2^2\right) d\tau. \tag{3}$$

Here, $\|.\|_2^2$ represents the Euclidean norm, and $\|.\|_1$indicates L_1 norms. The firing rule presented here represents that the spike emission is conditioned on the reduction of the predicted error, an analogy with predictive coding [31]. Therefore, utilizing such a rule, it can be ensured that neurons will only spike when they can contribute to the reduction of the cumulative error. On the other hand, it is notable that, for each individual neuron in the network, the membrane potential and thresholds can be considered using the following expressions:

$$\sigma_i(t) = D_i^T\left(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\right) - \mu r_i \tag{4}$$

$$T_i = \left(D_i^T D_i + \nu + \mu\right)/2. \tag{5}$$

In the aforementioned equations, σ_i is the membrane potential of the i^{th} spiking neuron, and T_i defines the thresholds for any individual neuron in the network, where D_i is the i^{th} column of matrix D, characterizing the neuron's output kernel and the change in the error due to a spike of the ith neuron. Thus, the neurons will emit spikes when their potential touches the threshold. Parameters μ and ν play critical roles in our SNN architecture. These parameters indeed serve crucial roles in maintaining network stability and preventing activity collapse in recurrent SNNs. Parameter μ addresses the tendency of recurrent networks to concentrate activity in a small subset of neurons by promoting balanced spike distribution across the network, preventing neuronal dominance patterns that lead to information bottlenecks. Meanwhile, parameter ν regulates overall network sparsity, creating an important balance between computational efficiency and sufficient neural activity for accurate control signal reproduction. Together, they form a homeostatic mechanism that ensures stable neural dynamics during prolonged operation—essential for consistent performance in our control application. After differentiating and simplification of (4), the following expression will be achieved:

$$\dot{\boldsymbol{\sigma}} = -\lambda\boldsymbol{\sigma} + D^T\boldsymbol{c} - \left(D^T D + \mu I\right)\boldsymbol{s} + D^T\left(\lambda\boldsymbol{x} + f(\boldsymbol{x})\right). \tag{6}$$

In the aforementioned expression, λ is the leak rate; also, the optimal weight matrix for the fast connections is defined by $\Omega_f = D^T\,D + \mu I$, and the optimal encoding matrix is D^T. Furthermore, the aforementioned expression says that the SNN can replicate any arbitrary dynamics like $\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + \boldsymbol{c}(t)$, based on the satisfaction of two assumptions. First, upon the inspiration from nonlinear adaptive control theories, for the last term, the approximation of $\lambda\boldsymbol{x} + f(\boldsymbol{x}) = \sum_i \Omega_i\phi_i(\boldsymbol{x})$ can be used, where Ω_i is the i^{th}column of Ω, and $\phi_i(.)$ can be any sigmoidal nonlinearity that here is set to $\tanh()$. Then, based on (1), the desired signal $\boldsymbol{x}$ can be replaced by $D\boldsymbol{r}$ that will lead to the following:

$$\dot{\boldsymbol{\sigma}} = -\lambda\boldsymbol{\sigma} + D^T\boldsymbol{c} - \Omega_f\boldsymbol{s} + D^T\Omega^T\phi(D\boldsymbol{r}). \tag{7}$$

From the biology perspective, the last term refers to the nonlinear and highly structured dendrites that perform nonlinear operations in the biological brains. Therefore, term $\phi(D\boldsymbol{r})$ is replaced by $\psi_i\ (\boldsymbol{r}) = \phi_i\ (M_i^T\boldsymbol{r} + \theta_i)$, which is a highly unstructured nonlinear dendrite that can receive stochastic inputs from other neurons, where M is the random fixed matrix, and θ_i is a parameter that determines where the tanh(.) changes its sign in a dendritic branch. It is notable that this substitution is accepted only when this substitution is justified by the fact that the internal dynamics meet the inequality $N > 2K$, because in such conditions, the network is highly robust to great amounts of noise [23], [26]. Then, network dynamics will be, thus, defined by the following expression:

$$\dot{\boldsymbol{\sigma}} = -\lambda\boldsymbol{\sigma} + F\boldsymbol{c} - \Omega_f\boldsymbol{s} + \Omega_s\psi(\boldsymbol{r}). \tag{8}$$

To apply a local learning rule inspired by adaptive control theories, the error $\boldsymbol{e}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$ is not propagated through the entire network via the input $\boldsymbol{c}(t)$. Instead, it is directly introduced to each neuron using the optimal encoder D^T. As a result, with these neuron-specific error feedbacks, the network's equation is modified as follows:

$$\dot{\boldsymbol{\sigma}} = -\lambda\boldsymbol{\sigma} - \Omega_f\boldsymbol{s} + \Omega_s\psi(\boldsymbol{r}) + kD^T\boldsymbol{e} \tag{9}$$

where

$$\dot{\Omega}_s = \eta\psi(\boldsymbol{r})\left(D^T\boldsymbol{e}\right)^T \tag{10}$$

In the aforementioned expressions, $\psi\ (\boldsymbol{r}) = \tanh(M^T\boldsymbol{r} + \boldsymbol{\theta})$ that can receive stochastic inputs from other neurons, where M is the random fixed matrix, and $\boldsymbol{\theta}$ is a random vector. Moreover, the aforementioned expression shows that the network utilizes two types of slow and fast synaptic connections: the slow connections Ω_s and the fast connections Ω_f. Slow connections control the main system behavior and dynamics, while fast connections ensure that spikes are distributed evenly throughout the network [26]. Finally, using the network design introduced here, the SNN-based controller at the edge can replicate the control signal $\boldsymbol{u}$ provided by the cloud server as the desired signal. Fig. 2 demonstrates the cloud-edge

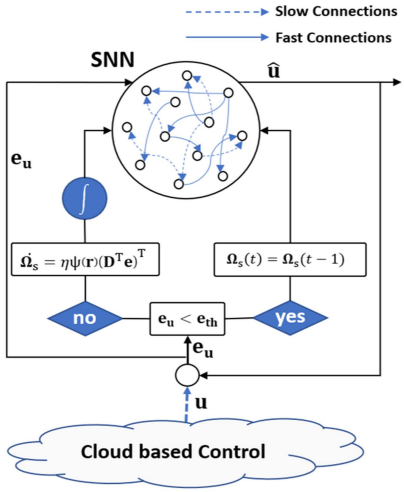

Fig. 2. Schematic diagram of weight update rule for SNN-based controllers at the edge [26].

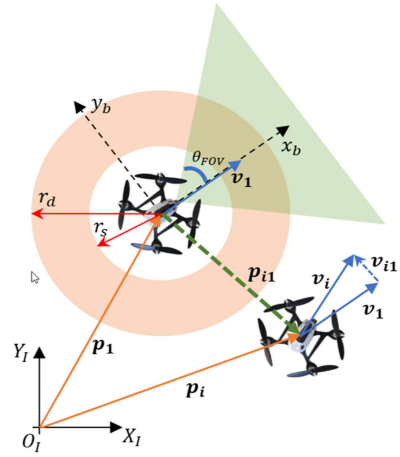

Fig. 3. Schematic design of the multi-UAV system [27].

communication framework and the process of network weight updates for the UAVs during the mission. By establishing a predefined threshold for prediction error $\boldsymbol{e}(t) = \boldsymbol{u}(t) - \hat{\boldsymbol{u}}(t)$, each UAV autonomously updates its weight matrix, independent of other UAVs, to accurately reproduce its control input signal $\hat{\boldsymbol{u}}$ [26].

In the introduced strategy in [26], for each UAV, the SNN-based edge controller undergoes two distinct phases of cloud communication. During the initial phase, spanning the first 50 time steps, the controller maintains continuous communication with the cloud. After this learning period, the system transitions to a more efficient operation mode, where cloud communication is significantly reduced. To maintain performance and prevent significant deviations from the desired control signal, the edge controller receives updates from the cloud every five time steps. During these periodic updates, the controller obtains the reference signal, updates its internal state, and performs threshold-based weight adjustments using the latest error calculations. This optimized communication scheme results in a huge reduction in energy consumption for cloud-edge data transfer compared to the initial phase, making it highly efficient for practical implementations.

B. Nature-Inspired Controller Design

This section focuses on adopting the nature-inspired collision-free control approach proposed in [27] to the indoor deployment of a multi-UAV system running on our proposed neuromorphic DT-based architecture. Fig. 3 demonstrates the schematic design of a multi-UAV system. In the presented design, to model the inertial translational motion of UAVs and their local dynamics, inertial coordinate frame and body coordinate frames represented by $O_I X_I Y_I$ and $O_b x_b y_b$ have been leveraged, respectively.

Considering the defined parameters in Fig. 3 and introducing $\boldsymbol{u}_i$ as to the auxiliary control input for the i^{th} UAV, the following dynamical equations are obtained [27]:

$$\dot{\boldsymbol{p}}_i = \boldsymbol{v}_i \tag{11}$$

$$\dot{\boldsymbol{v}}_i = \boldsymbol{u}_i \tag{12}$$

where $\boldsymbol{p}_i$ and $\boldsymbol{v}_i$ represent the position and velocity vectors of the i^{th} UAV, respectively. Parameters $\boldsymbol{p}_{ij} = \boldsymbol{p}_j - \boldsymbol{p}_i$ and $\boldsymbol{v}_{ij} = \boldsymbol{v}_j - \boldsymbol{v}_i$ are relative position and velocity vectors, respectively. r_s and r_d are safety range (none of the UAVs are allowed to get closer to other UAVs than this range) and detection range, respectively. θ_{FOV} is the field of view (FOV) of the UAV with respect to the flight direction. Also, as it is common, to mathematically model a multi-UAV system for the purposes such as formation controller design and obstacle avoidance maneuvers, the UAVs in this study have been considered particles [4], [33], [27].

To design the desired formation control law for the optimal deployment of a multi-UAV system consisting of N vehicles in a bounded area of space, the graph theory has been utilized. Thus, the graph $G(U, E)$ models a homogeneous system with a set of nodes defined as $U = \{i\}\ \ i = 1, 2, \ldots, n$, and a set of $E = \{(i, j) |\ i,\ j \in U;\ i \neq j\}$ that represents the edges that model the communication between the UAVs. Moreover, the following set of nodes in the space describes the neighborhood of each UAV in its detection range

$$N_i = \{j \in U \backslash \{i\}\ |\ ||\boldsymbol{p}_i - \boldsymbol{p}_j|| < r_d\}. \tag{13}$$

Furthermore, as introduced in [27], [34], [35], and [36], inspired by tilapia fish territorial behavior, for optimal deployment of UAVs in bounded spaces, a centroidal Voronoi tessellation (CVT) is required. This nature-inspired approach provides inherent adaptability to dynamic environments, particularly effective where UAVs must coordinate while avoiding obstacles and each other. As conditions change, the CVT algorithm automatically redistributes spatial territories, maintaining both coverage and safety. The multilayered control structure enables UAVs to temporarily deviate around obstacles without compromising swarm integrity. By integrating collision avoidance forces, CVT-based positioning, and a unified control law, the system adapts to obstacles and recovers its configuration autonomously, as demonstrated in our simulations (see Section III-B). This capability is essential for real-world complex environments.

This capability is critical for real-world applications in complex environments. To implement this nature-inspired control strategy mathematically, the UAVs need to obey the following

TABLE I
PSEUDOCODE OF THE IMPLEMENTED LLOYD'S CVT ALGORITHM

Probabilistic Generalized Lloyd's Algorithm

Input:

- Predefined bounded area $Q \subset R$
- Density function $\rho(x)$ defined on Q
- Positive integer N (number of UAVs)
- Positive integer S_{num} (number of sampling points per iteration)
- Constants $\alpha_1, \alpha_2, \beta_1$, and β_2 such that $\alpha_1 + \alpha_2 = 1$, $\beta_1 + \beta_2 = 1$, $\alpha_2 > 0$, $\beta_2 > 0$

Steps:

1. Initialization:

- Choose an initial set of N points $\{x_i\}_{i=1}^{N}$ in Q.
- Set iteration counters $\{j_i\}_{i=1}^{N} = 1$.

2. Sampling:

- Randomly sample S_{num} points $\{y_r\}_{r=1}^{S_{num}}$ in Q using a *uniform distribution* $\rho(x)$ as the probability density function.

3. Point Update:

For each i = 1, 2, ..., N:

- Gather all sampled points y_r closest to x_i (forming the set W_i, i.e., the Voronoi region of x_i).
- If W_i is empty, do nothing.
- Otherwise:
  - Compute the average u_i of the points in W_i.
  - Update x_i:
    $x_i \leftarrow ((\alpha_1 j_i + \beta_1) / (j_i + 1))\, x_i + ((\alpha_2 j_i + \beta_2) / (j_i + 1))\, u_i$
  - Increment j_i:
    $j_i \leftarrow j_i + 1$

4. Repeat or Terminate:

- Form the new set of points $\{x_i\}_{i=1}^{N}$.
- If Q is a hypersurface, project x_i onto Q
- Check stopping criteria (e.g., convergence or tolerance).
- If criteria are not met, go back to Step 2.

Output:

- Final set of points $\{x_i\}_{i=1}^{N}$.

control law [27]:

$$\boldsymbol{u}_{f_i} = -K_{p_i}\ (\boldsymbol{p}_i - \boldsymbol{c}_{v_i})\,. \tag{14}$$

In the aforementioned expression, K_{p_i} is a positive-definite gain matrix that can be designed by trial and error, and $\boldsymbol{c}_{v_i}$ refers to the centroid of each section in a CVT. Thus, to find the desired centroids for UAVs in a predefined bounded area, the probabilistic generalized Lloyd's method has been leveraged. The pseudocode of the implemented algorithm has been provided in Table I [27].

It is notable that for the presented study and the presented algorithm in Table I, the considered probability density function $\rho(x)$ is a uniform distribution, which will lead to a uniformly distributed configuration of UAVs. Furthermore, for considering the safety factors in the deployment of the multi-UAV system to avoid intervehicle collision and obstacle avoidance, the following parts have been added to the control law [27].

1) Collision Avoidance Controller

Inspired by Hook's law in the spring-mass-damper system and considering relative position and velocities between the vehicles,

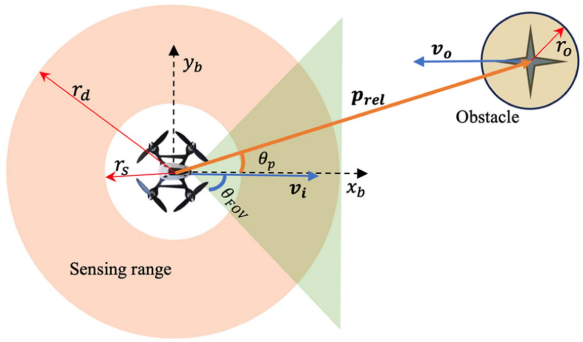

Fig. 4. Obstacle detection schematic representation [26].

the following expression has been introduced for the collision avoidance control law [27]:

$$\boldsymbol{u}_{cij} = -k_{c1}\boldsymbol{p}_c + k_{c2}\boldsymbol{v}_{ij} \tag{15}$$

where

$$\boldsymbol{p}_c = \frac{1}{\left(||\boldsymbol{p}_{ij}|| - r_s\right)^2}\,\frac{\boldsymbol{p}_{ij}}{||\boldsymbol{p}_{ij}||}. \tag{16}$$

In the aforementioned expressions, k_{c1} and k_{c2} are positive-definite matrices.

2) Obstacle Avoidance Controller

This section builds upon the obstacle avoidance controller design introduced in [27], which was inspired by how pigeons collectively maneuver around obstacles. We first review the original 2-D planar model, where obstacles were limited to movement in the x-direction. We then extend this approach to 3-D space, which is the primary focus of our research. The method consists of two main components: obstacle detection and maneuver execution. Fig. 4 shows the schematic representation of the obstacle detection.

Considering the sector-like FOV for the vehicles, and a circular sensing range, the two considerations have to be met. First, the obstacle has to be in the sensing range; second, it has to be in the FOV of the vehicles. Then, the obstacle avoidance needs to be performed. To this aim, the algorithm presented in Table II needs to be utilized.

The presented algorithm in Table II shows that in this study, for the obstacle avoidance maneuver, forbidden areas around the obstacles have been considered. Thus, for the overall control law of the vehicles, the following expression has to be implemented:

$$\boldsymbol{u}_i = \boldsymbol{u}_{fi} + \boldsymbol{u}_{ci}. \tag{17}$$

Through the aforementioned expression, each UAV obtains its control input vector in a semidistributed fashion, where Lloyd's algorithm defines the desired spatial positions of the UAVs.

III. NUMERICAL SIMULATION

This section presents the obtained results from the implementation of the neuromorphic DT framework into the indoor deployment and a collision-free formation control scenario of a multi-UAV system considering obstacle avoidance.

TABLE II
PSEUDOCODE OF OBSTACLE DETECTION AND AVOIDANCE

Obstacle detection and avoidance algorithm

 # Initialize obstacle detection metric
$OD_{metric_{range}}$ = Array of zeros with size of number of obstacles K
Loop over obstacles
FOR i = 1 TO K DO:
  # Calculate distance to the obstacle
  $OD_{metric_{range}}$ = Euclidean norm of $(p_j - p_{Obs_i})$ for i^{th} obstacle with respect to j^{th} vehicle
  # Calculate FOV
  $OD_{metric_{FOV}}$ = Absolute difference of angles between object-obstacle and velocity direction
  # Check if obstacle is detected
  IF ($OD_{metric_{range}}$ < (obstacle radius + r_d)) AND ($OD_{metric_{FOV}} < \theta_{FOV}$):
    Set OD_{metric}[i] = 1
  ELSE:
    Set OD_{metric}[i] = 0
  ENDIF
  # While any obstacle is detected
  WHILE sum(OD_{metric}) > 0 DO:
    # Adjust position to avoid obstacle
    IF $p_y > p_{Obs_{i_y}}$:
      p_x = Reduce magnitude of p_x by *scaler*
      p_y = Increase magnitude of p_y by *scaler*
    ELSE:
      p_x = Reduce magnitude of p_x by *scaler*
      p_y = Reduce magnitude of p_y by *scaler*
    ENDIF
  Update p to new adjusted values (p_x,p_y , and original p_z)
    # Recalculate range and FOV metrics
    Update $OD_{metric_{range}}$ and $OD_{metric_{FOV}}$ with new values
    # Check if obstacle is still detected
  IF ($OD_{metric_{range}}$ < (obstacle radius + r_d)) AND ($OD_{metric_{FOV}} < \theta_{FOV}$):
    Set OD_{metric}[i] = 1
  ELSE:
    Set OD_{metric}[i] = 0
  ENDIF
  ENDWHILE
ENDFOR

A. Case Study 1: Indoor Deployment of the Multi-UAV System

This section presents the simulation results for deploying 15 UAVs in a rectangular area, approximating an indoor surveying scenario. Simulations were run for 10 s with a 0.01-s time step, using the numerical values in Table III. Throughout this work, the elements of matrix D are sampled from a zero-mean Gaussian distribution with covariance 1.

Fig. 5 shows the UAVs' optimal configuration in the enclosed space, computed via Lloyd's method. Each Voronoi partition's center serves as a target position for the UAVs' controller.

Fig. 6 shows the deployment results of a multi-UAV system using our proposed neuromorphic DT-based control architecture. It compares the predicted trajectories from a virtual twin—receiving control signals directly from the cloud-based controller—to those generated by an edge device model using an SNN-based controller. The alignment of both trajectories indicates that the SNN-based approach at the edge successfully learns and applies the control signals expected from the virtual twin.

TABLE III
NUMERICAL VALUES FOR THE SIMULATIONS—INDOOR DEPLOYMENT

Parameter	Value
$r_d(m)$	3
$r_s(m)$	1
K_p	$diag([1,1,1])$
K_v	$diag([2,2,3.5])$
k_{c1}	10
k_{c2}	1
θ_{FOV}(deg)	60
μ	0.0001
λ	0.0001
ν	0.0001
k	500
η	0.01

TABLE IV
RESOURCE UTILIZATION OF THE UAVS—INDOOR DEPLOYMENT

UAV	Number of Spikes	Resource Utilization (%)
1	6110	6.1
2	7745	7.7
3	3060	3.1
4	1649	1.7
5	1700	1.7
6	2257	2.3
7	2266	2.3
8	2154	2.2
9	1328	1.3
10	2572	2.6
11	3258	3.3
12	2670	2.7
13	4801	4.8
14	3117	3.1
15	3849	3.8

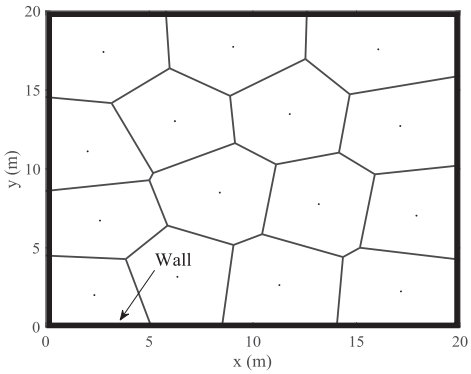

Fig. 5. Optimal formation of UAVs in the considered area obtained from Lloyd's method.

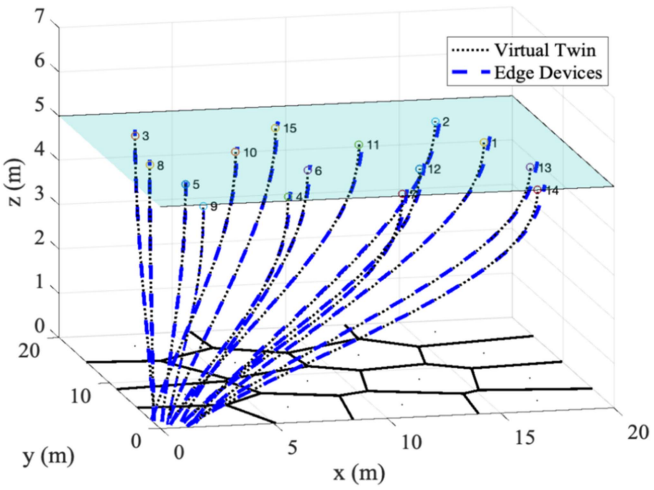

Fig. 6. Obtained trajectories for UAVs from virtual twins compared with obtained trajectories for edge devices utilizing signal reproduced by the SNN-based controller. Video and Code are available online at https://github.com/INQUIRELAB/SNN-Based-UAV-Formation-Control.

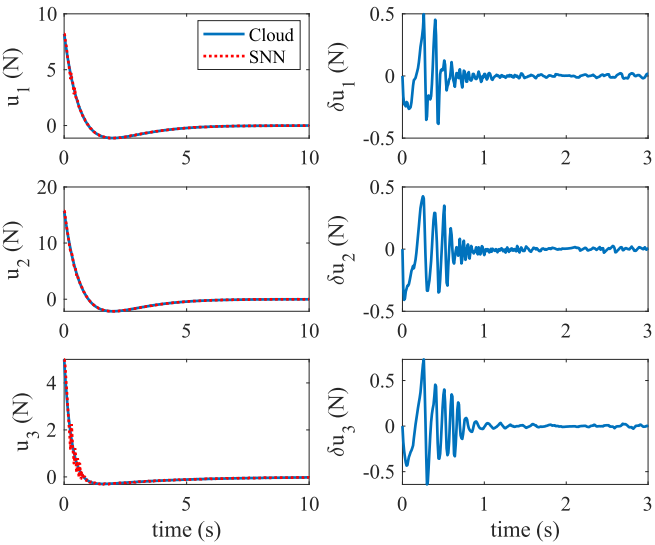

Fig. 7. Comparison of the obtained control inputs from the controller on the cloud with the reproduced control signals by the SNNs at the edge accompanied with their corresponding errors for the first 3 s of simulation.

Fig. 7 provides a more detailed comparison between the cloud-based controller's outputs and the SNN-based reproduction for UAV15. The SNN effectively replicates the desired control input, as evidenced by tracking errors converging to zero before $t = 1$ s with negligible fluctuations. This demonstrates the SNN's ability to learn and reproduce the control signals in an event-driven manner on edge devices.

Fig. 8 illustrates the spiking pattern for UAV15. Most neural activity occurs before $t = 1$ s (i.e., the first 100 time steps), corresponding to the interval during which the tracking errors in Fig. 7 converge to zero. After that point, the network remains largely idle, reflecting the event-based nature of SNNs, which become active only when significant deviations exist

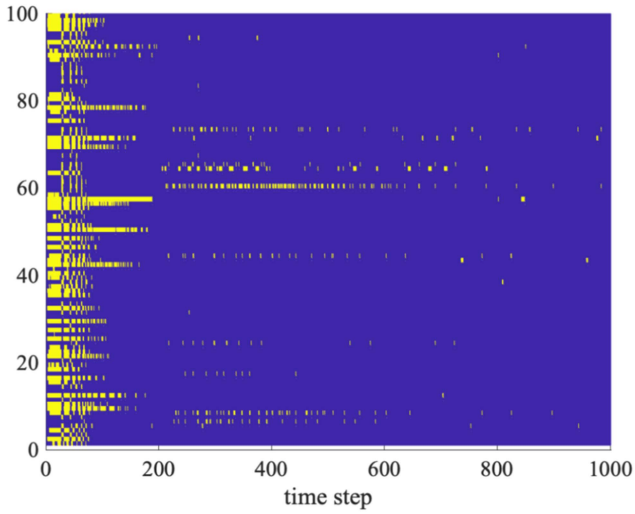

Fig. 8. Obtained spiking pattern for the SNN-based controller considered for UAV15 during the deployment scenario.

between the reproduced and actual control signals. Quantitatively, the UAV15 controller uses 3849 spikes—about 3.85% of the 100 000 available—confirming that SNNs reduce computational effort once tracking errors are minimal.

It is worth noting that, because the results for all UAVs are qualitatively similar, only the data for UAV15 (chosen randomly) are presented here. To further analyze the proposed method's computational efficiency in a large-scale system of 15 UAVs, Table IV provides the corresponding quantitative results. This table shows that the resource consumption for all UAVs ranges between approximately 1% and 8%, confirming the efficiency of our approach. In this architecture, the SNN-based controllers operate entirely independently of the cloud-based algorithm, eliminating the need to train separate flight computers on every controller parameter. Consequently, implementing our proposed architecture requires only minimal resources at the edge devices—an advantage in distributed multi-UAV operations.

B. Case Study 2: Moving a Flock of UAVs With Obstacles

This section presents the obtained results from the implementation of our proposed architecture in the collision-free control of a moving flock of UAVs consisting of six UAVs considering obstacle detection and avoidance maneuvers. Simulation here has been conducted using the numerical values provided in Table V for 50 s with a time step of 0.01 s.

In the considered scenario here, first, the UAVs will form an optimal configuration obtained from Lloyd's method in an area defined by $\{(x, y, z) | \, 0 < x < 2; 0 < y < 4; z = 5\}$; then, they will move in the x-direction, while they encounter two static obstacles. Fig. 9 illustrates the obtained optimal configuration for the UAV flock in the desired area. Each point in the centroid of the Voronoi partitions has been assigned to any individual UAV.

Fig. 10 compares the trajectories produced by the virtual twin model to those generated by the edge device models,

TABLE V
NUMERICAL VALUES FOR THE SIMULATIONS—MOVING FLOCK

Parameter	Value
$r_d(m)$	1.5
$r_s(m)$	0.5
K_p	$diag([1,1,1])$
K_v	$diag([2,2,2])$
k_{c1}	3
k_{c2}	1
θ_{FOV}(deg)	60
μ	0.001
λ	0.005
ν	0.001
k	500
η	0.001

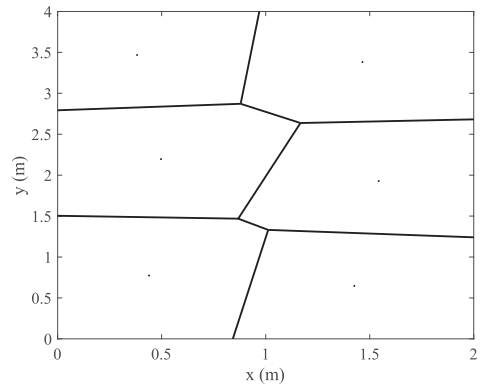

Fig. 9. Optimal configuration for the UAVs from Lloyd's method in the moving flock scenario.

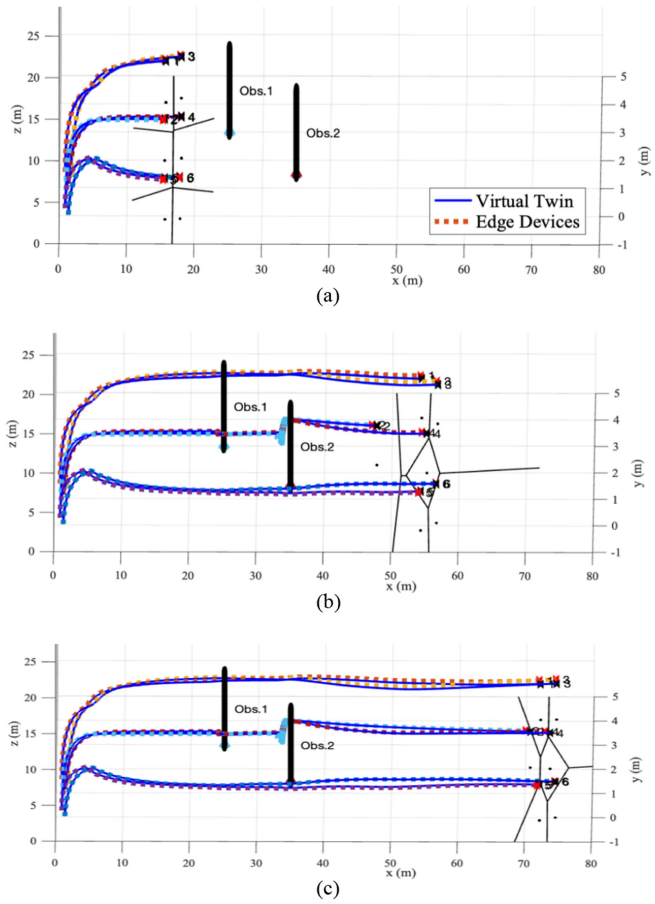

Fig. 10. (a)–(c) obtained trajectories for UAVs from virtual twin compared with obtained trajectories for edge devices utilizing signals reproduced by the SNN-based controllers for the moving UAV flock.

which rely on SNN-based control signals. Overall, the plots illustrate how the edge devices successfully track the expected paths dictated by the virtual twin. Specifically, Fig. 10(a) shows the first phase of the scenario in which the UAVs converge to their optimal configuration (from Lloyd's method) with no intervehicle collisions. Fig. 10(b) captures the second phase, where the vehicles detect and avoid two separate obstacles, yet continue following their expected trajectories. In Fig. 10(c), the UAV formation recovers its optimal configuration without collisions, confirming the viability of the proposed architecture for multi-UAV surveying or mapping tasks. It is worth noting that, under closer scrutiny, the edge device trajectories exhibit minor deviations from those of the virtual twin. This indicates a small loss of accuracy in exchange for the computational efficiency of the SNN-based approach—an acceptable tradeoff given the size of the surveyed area.

Fig. 11 shows the spiking patterns for all UAVs in the moving flock. Although most networks display similar activity, UAVs 2 and 4—each encountering an obstacle in the x-direction—exhibit distinctly different behavior. After time step 2000, their neural firing increases sharply for a brief interval before returning to near-idle levels, whereas other UAVs remain largely idle throughout. This demonstrates the event-based nature of the SNN-based controllers, which adjust computational load and resource usage according to significant changes in input data. In contrast, a traditional approach, such as an ANN, would remain fully active at every time step, continuously consuming 100% of available resources. Moreover, in our architecture, each SNN operates independently, so elevated neural activity from UAVs 2 and 4 does not affect the other networks.

TABLE VI
RESOURCE UTILIZATION OF UAVS—MOVING FLOCK

UAV	Number of Spikes	Resource Utilization (%)
1	33 730	4.5
2	73 583	9.8
3	31 362	4.2
4	39 015	5.2
5	26 067	3.5
6	23 304	3.1

To qualitatively assess the resource consumption of UAVs in this scenario, Table VI lists the number of spikes used by each SNN-based controller to reproduce the desired control signals. Given the SNN size and total simulation time, up to 750 000 spikes are available. As shown in Table VI, UAVs that did not encounter obstacles used between 3% and 4.5% of their

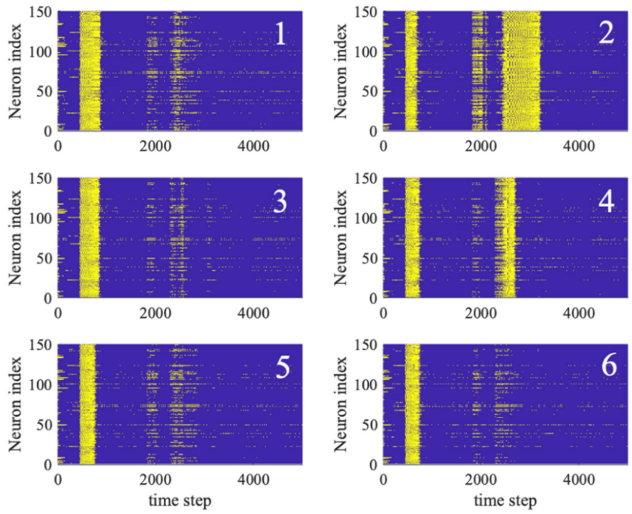

Fig. 11. Obtained spiking patterns for the UAVs in the moving flock.

possible spikes, whereas UAVs 2 and 4—both of which interacted with obstacles—consumed around 10%. These findings quantitatively confirm the spiking pattern observations, where additional neural activity arises in response to obstacles, leading to higher resource usage. Our architecture specifically addresses operation during communication interruptions through a robust learning and adaptation mechanism. During the initial learning phase (first 50 time steps as described in Section II-A), each UAV's SNN develops a comprehensive internal representation of the control policy by learning to reproduce cloud-generated control signals. This representation is encoded primarily in the slow connection weights (Ω_s) as defined in (10), creating a neural model of the desired control behavior. After this initial learning phase, the SNNs demonstrate remarkable resilience to communication disruptions. The networks can autonomously generate appropriate control signals without continuous cloud connectivity, receiving only periodic updates every five time steps to maintain alignment with the cloud controller. This designed intermittency confirms the system's inherent robustness to communication gaps—a critical advantage in real-world deployment scenarios where connectivity cannot be guaranteed.

The spiking patterns visualized in Figs. 8 and 11 illustrate another key efficiency of our approach: computational activity concentrates during significant control signal transitions while remaining minimal during stable flight phases. This event-driven processing conserves computational resources when control demands are low. Furthermore, as evidenced by (10), weight updates depend on both the prediction error and current neural activity, enabling the network to continuously refine its control policy even during periods of limited communication. During complete communication interruptions, the network leverages its most recently updated error values and current filtered spike trains to maintain effective control based on its learned policy.

IV. Conclusion

The proposed distributed cloud-edge architecture for multi-UAV systems offers a transformative approach to addressing the challenges of task complexity, energy efficiency, and real-time adaptation in dynamic and obstacle-rich environments. By integrating SNNs with nature-inspired control strategies, the framework achieves a robust balance between local autonomy and the computational capabilities of cloud-based controllers. The architecture's ability to maintain effective operation during temporary cloud connectivity losses demonstrates its resilience and adaptability, while its use of neuromorphic computing ensures energy efficiency and low-latency performance.

Experimental results in two distinct scenarios—collision-free indoor deployment of 15 UAVs and formation control of a moving flock of six UAVs—highlight the framework's effectiveness in achieving optimal collision-free operations in complex environments. The synergy between SNN-based local controllers and nature-inspired cloud-driven global strategies enables scalable decentralized control that can adapt to dynamic conditions in real time. This approach represents a significant step forward in the design of multiagent autonomous systems, particularly for applications in urban and other complex operational domains. Future work may focus on further optimizing the framework's scalability and exploring its potential in other multiagent contexts.

References

[1] J. Tang, G. Liu, and Q. Pan, "A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 10, pp. 1627–1643, Oct. 2021.

[2] C. D. Schuman, S. R. Kulkarni, M. Parsa, J. P. Mitchell, P. Date, and B. Kay, "Opportunities for neuromorphic computing algorithms and applications," *Nat. Comput. Sci.*, vol. 2, no. 1, pp. 10–19, 2022.

[3] R. J. A. A. Nathan, I. Krumi, and O. Bimber, "Drone swarm strategy for the detection and tracking of occluded targets in complex environments," *Commun. Eng.*, vol. 2, no. 1, 2023, Art. no. 55.

[4] J. Guo et al., "Distributed cooperative obstacle avoidance and formation reconfiguration for multiple quadrotors: Theory and experiment," *Aerosp. Sci. Technol.*, vol. 136, 2023, Art. no. 108218.

[5] M. S. Muhmoud, "Cloud-based control systems: Basics and beyond," *J. Phys.: Conf. Ser.*, vol. 1334, no. 1, 2019, Art. no. 012006.

[6] J. Schlechtendahl, F. Kretschmer, Z. Sang, A. Lechler, and X. Xu, "Extended study of network capability for cloud based control systems," *Robot. Comput.-Integr. Manuf.*, vol. 43, pp. 89–95, 2017.

[7] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, 1997.

[8] S. Huang and R. S. H. Teo, "Computationally efficient visibility graph-based generation of 3D shortest collision-free path among polyhedral obstacles for unmanned aerial vehicles," in *Proc. Int. Conf. Unmanned Aircr. Syst.*, 2019, pp. 1218–1223.

[9] Y. I. Jenie, E. J. Kampen, C. C. de Visser, J. Ellerborek, and J. M. Hoekstra, "Selective velocity obstacle method for deconflicting maneuvers applied to unmanned aerial vehicles," *J. Guid., Control, Dyn.*, vol. 38, no. 6, pp. 1140–1146, 2015.

[10] F. Bounini, D. Gingras, H. Pollart, and D. Gruyer, "Modified artificial potential field method for online path planning applications," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 180–185.

[11] E. Soria, F. Schiano, and D. Floreano, "Predictive control of aerial swarms in cluttered environments," *Nat. Mach. Intell.*, vol. 3, no. 6, pp. 545–554, 2021.

[12] Y. Rasekhipour, A. Khajepour, S. K. Chen, and C. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1255–1267, May 2017.

[13] Z. Yu, Z. Si, X. Li, D. Wang, and H. Song, "A novel hybrid particle swarm optimization algorithm for path planning of UAVs," *IEEE Internet Things J.*, vol. 9, no. 22, pp. 22547–22558, Nov. 2022.

[14] H. Chu, J. Yi, and F. Yang, "Chaos particle swarm optimization enhancement algorithm for UAV safe path planning," *Appl. Sci.*, vol. 12, no. 18, 2022, Art. no. 8977.

[15] Q. Yao et al., "Path planning method with improved artificial potential field—A reinforcement learning perspective," *IEEE Access*, vol. 8, pp. 135513–135523, 2020.
[16] Z. Pan, C. Zhang, Y. Xia, H. Xiong, and X. Shao, "An improved artificial potential field method for path planning and formation control of the multi-UAV systems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 3, pp. 1129–1133, Mar. 2022.
[17] G. Hao, Q. Lv, Z. Huang, H. Zhao, and W. Chen, "UAV path planning based on improved artificial potential field method," *Aerospace*, vol. 10, no. 6, 2023, Art. no. 562.
[18] F. Kong, Q. Wang, S. Gao, and H. Yu, "B-APFDQN: A UAV path planning algorithm based on deep Q-network and artificial potential field," *IEEE Access*, vol. 11, pp. 44051–44064, 2023.
[19] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "Enhancing energy efficiency and reliability in autonomous systems estimation using neuromorphic approach," 2023, *arXiv:2307.07963*.
[20] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "Neuromorphic robust estimation of nonlinear dynamical systems applied to satellite rendezvous," *Adv. Space Res.*, vol. 75, no. 3, pp. 3010–3024, 2025.
[21] H. T. Lin, I. G. Ros, and A. A. Biewener, "Through the eyes of a bird: Modelling visually guided obstacle flight," *J. Roy. Soc. Interface*, vol. 11, no. 96, 2014, Art. no. 20140239.
[22] M. Huo, H. Duan, Q. Yang, D. Zhang, and H. Qiu, "Live-fly experimentation for pigeon-inspired obstacle avoidance of quadrotor unmanned aerial vehicles," *Sci. China Inf. Sci.*, vol. 62, pp. 1–8, 2019.
[23] A. Alemi, C. Machens, S. Deneve, and J. J. Slotine, "Learning nonlinear dynamics in efficient, balanced spiking networks using local plasticity rules," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1. [Online]. Available: https://doi.org/10.1609/aaai.v32i1.11320
[24] M. Zhang, K. Ma, L. Liu, and W. Zhang, "Multiobjective spherical vector-based particle swarm optimisation for 3D UAV path planning," in *Proc. 6th Int. Conf. Data-Driven Optim. Complex Syst.*, 2024, pp. 427–432.
[25] M. Abdel-Basset, R. Mohamed, K. M. Sallam, I. M. Hezam, K. Munasinghe, and A. Jamalipour, "A multiobjective optimization algorithm for safety and optimality of 3-D route planning in UAV," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 60, no. 3, pp. 3067–3080, Jun. 2024.
[26] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "A cloud-edge framework for energy-efficient event-driven control: An integration of online supervised learning, spiking neural networks and local plasticity rules," *IoP Sci. Neuromorphic Comput. Eng.*, vol. 4, no. 4, 2024, Art. no. 044004.
[27] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "Swarm intelligence in collision-free formation control for multi-UAV systems with 3D obstacle avoidance maneuvers," 2024, *arXiv:2412.12437*.
[28] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "Neuromorphic digital-twin for multi-agent system control using spiking neural networks," in *Proc. Conf. AI, Sci., Eng., Technol.*, 2024, pp. 204–207.
[29] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "Advancing precision in multi-agent systems: A neuromorphic approach with spiking neural network-modified sliding innovation filter," *Proc. SPIE*, vol. 12949, 2024, Art. no. 129490T.
[30] G. Tang, "Biologically inspired spiking neural networks for energy-efficient robot learning and control," Ph.D. dissertation, School Graduate Stud., Rutgers The State Univ. New Jersey, New Brunswick, NJ, USA, 2022.
[31] M. Boerlin, C. K. Mchanes, and S. Deneve, "Predictive coding of dynamical variables in balanced spiking networks," *PLoS Comput. Biol.*, vol. 9, no. 11, pp. e10032–e10058, 2013.
[32] F. S. Slijkhuis, S. W. Keemink, and P. Lanillos, "Closed-form control with spike coding networks," *IEEE Trans. Cogn. Develop. Syst.*, 2023.
[33] T. Elmokadem and A. V. Savkin, "Computationally-efficient distributed algorithms of navigation of teams of autonomous UAVs for 3D coverage and flocking," *Drones*, vol. 5, no. 4, 2021, Art. no. 124.
[34] J. Cortes, S. Martinez, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Autom.*, vol. 20, no. 2, pp. 243–255, Apr. 2004.
[35] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 401–420, Mar. 2006.
[36] A. Cortes, S. Martinez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM: Control, Optim. Calculus Variations*, vol. 11, no. 4, pp. 691–719, 2005.