Memorias del XX Congreso Latinoamericano de Control Automático (CLCA 2025)
13-17 de Octubre, 2025. Cancún, Quintana Roo, México

Modular Evolution of Artificial Potential Fields for Coordinated Aerial Navigation ⋆

J. J. Cetina-Denis [*] **C. Cruz-Hernández** [*] **M. A. Chan-Ley** [**]
A. Arellano-Delgado [***,b] [***] **M. A. Murillo-Escobar** [b] [***]

[*] *Applied Physics Division, Center for Scientific Research and Higher Education at Ensenada, Ensenada, Baja California, 22860, México (e-mail: jcetina@cicese.edu.mx, ccruz@cicese.mx).*
[**] *Nanoscience and Nanotechnology Center, National Autonomous University of Mexico, Ensenada, Baja California, 22860, México (e-mail: mchan@ens.cnyn.unam.mx)*
[***] *Secretariat of Science, Humanities, Technology, and Innovation (SECIHTI), Av. de los Insurgentes Sur 1582, Ciudad de México, 03940, México,*
[b] [***] *Engineering, Architecture, and Design Faculty, Autonomous University of Baja California, Ensenada, 22860, Baja California, México (e-mail: adrian.arellano@uabc.edu.mx, murillo.miguel@uabc.edu.mx)*

Abstract: Artificial Potential Fields (APFs) are commonly used to control multi-agent systems such as drone swarms, enabling behaviors like trajectory tracking, coordination, and collision avoidance. However, APFs are typically hand-crafted and require extensive tuning, limiting their adaptability. To address this, we propose an evolutionary strategy based on Genetic Programming (GP) to automatically synthesize vector-valued controllers that replace traditional APF components for attraction, synchronization, and repulsion. Controllers are evolved in a 3D quadrotor simulation with realistic dynamics and evaluated on position and velocity errors during coordinated trajectory tracking. Results show that the evolved controllers match or outperform classical APFs, enabling emergent coordination without manual design. This demonstrates the potential of evolutionary synthesis for scalable, adaptive multi-agent control.

Keywords: Autonomous Mobile Robots, UAVs, Evolutionary Algorithms, Decentralized Control, Trajectory Tracking and Path Following

1. INTRODUCTION

Coordinating multiple aerial robots is a central challenge in swarm robotics, with applications in monitoring, agriculture, and disaster response (Debie et al. (2023); Arnold et al. (2018)). Artificial potential fields (APFs) are commonly used for decentralized control by encoding attractive and repulsive forces (Klančar et al. (2022); Sfeir et al. (2011)). While effective, classical APFs require manual tuning and suffer from issues like local minima (Kam (2017)).

To overcome these limitations, prior work has introduced enhancements such as adaptive fields, velocity alignment, and formation schemes (see Huang et al. (2019); Selvam et al. (2021); Fujimori et al. (2014); Handayani et al. (2015)). However, most rely on fixed structures that limit adaptability. Genetic programming (GP) offers a flexible alternative by evolving control laws from simulation or feedback, as seen in Horváth et al. (2021); Meza-Sánchez et al. (2019); Neupane and Goodrich (2019); Besada et al. (2010); li Su and Wang (2021), though many efforts ignore physics or constrain the solution space.

We propose a method to evolve interpretable vector-valued APF controllers using strongly typed GP for quadrotor swarms. The evolved expressions synthesize attraction and synchronization terms from kinematic inputs and are evaluated in a full 3D simulation with realistic quadrotor dynamics. Unlike traditional APFs, our approach searches over symbolic expressions, enabling novel coordination strategies. Results show improved multi-agent trajectory tracking and synchronized motion, demonstrating potential for scalable, adaptive control in aerial swarms.

2. ARTIFICIAL POTENTIAL FIELDS FOR MOBILE ROBOT CONTROL

Artificial Potential Fields (APFs) are a popular framework for decentralized control in multi-agent and swarm robotics. Each agent computes its control input as the sum of virtual forces derived from potential functions: an *attractive* force toward the goal, and *repulsive* forces to avoid collisions or restricted areas (Khatib (1986)). A *synchronization* force can also be included to promote coherent motion.

⋆ This work was supported by the Secretariat of Science, Humanities, Technology and Innovation (SECIHTI).

https://doi.org/10.58571/CNCA.AMCA.2025.069

The overall control law for agent i is:

$$\mathbf{u}_i = \mathbf{f}_{\text{att}}(\mathbf{p}_i, \mathbf{p}_g) + \mathbf{f}_{\text{rep}}(\mathbf{p}_i, \mathbf{p}_g) + \mathbf{f}_{\text{sync}}(\mathbf{v}_i, \mathbf{v}_g), \quad (1)$$

where $\mathbf{p}_i, \mathbf{v}_i$ are the position and velocity of agent i and $\mathbf{p}_g$ and $\mathbf{v}_g$ are the goal position and velocity.

Attraction Potentials

Common attraction functions include:

- **Quadratic:** $\mathbf{f}_{\text{att}} = -k_a(\mathbf{p}_i - \mathbf{p}_g)$
- **Conic:** $\mathbf{f}_{\text{att}} = -k_a \cdot \frac{(\mathbf{p}_i - \mathbf{p}_g)}{\|\mathbf{p}_i - \mathbf{p}_g\| + \epsilon}$
- **Logarithmic:** $\mathbf{f}_{\text{att}} = -k_a(\mathbf{p}_i - \mathbf{p}_g) \cdot \log(1 + \|\mathbf{p}_i - \mathbf{p}_g\| + \epsilon)$

where $\mathbf{p}_i$ and $\mathbf{p}_g$ are the agents and target position respectively, k_a is the attraction gain that dictates how strong of a pull the agent feels towards the target, and ϵ is a small positive constant used to prevent division by zero or the logarithm of zero.

Velocity Synchronization

To ensure smooth group motion, agents can reduce velocity mismatches:

- **Linear damping:** $\mathbf{f}_{\text{sync}} = -k_s(\mathbf{v}_i - \mathbf{v}_g)$
- **Exponential decay:** $\mathbf{f}_{\text{sync}} = -k_s(\mathbf{v}_i - \mathbf{v}_g)\left(1 - e^{-\alpha\|\mathbf{v}_i - \mathbf{v}_g\|}\right)$
- **Saturation:** $\mathbf{f}_{\text{sync}} = -k \cdot \text{clip}(\mathbf{v}_i - \mathbf{v}_g, -v_{\max}, v_{\max})$

where $\mathbf{v}_i$ is the agent's current velocity, and $\mathbf{v}_g$ is the desired target velocity, the gain k_s determines how strongly the agent aligns its velocity with the target, and α controls how quickly the alignment force grows with increasing velocity error. These are widely used in UAV tracking to improve convergence and minimize overshoot (Bouček et al. (2024); Zhao et al. (2023)).

Repulsion and Formation Control

Repulsive forces prevent collisions and help maintain formation:

- **Linear:** $\mathbf{f}_{\text{rep}} = k_r(\mathbf{p}_i - \mathbf{p}_j)$
- **Inverse-distance:** $\mathbf{f}_{\text{rep}} = \frac{k_r(\mathbf{p}_i - \mathbf{p}_j)}{d + \epsilon}$
- **Planar spacing (flocking-inspired):** $\mathbf{f}_{\text{rep}} = k_r \cdot \frac{\mathbf{p}_i - \mathbf{p}_j}{\|\mathbf{p}_i - \mathbf{p}_j\|}(d_0 - \|\mathbf{p}_i - \mathbf{p}_j\|)$

where k_r is a gain and d_0 is the desired inter-agent distance (Olfati-Saber (2006); Ren and Beard (2005)).
$\mathbf{f}_{\text{rep},z} = 0$ allows lateral separation without disrupting vertical coordination.

Despite their simplicity and real-time capabilities, classical APFs require careful design and tuning. They can suffer from local minima, oscillations, and poor scalability in dynamic or dense swarms Rossi et al. (2018); Qin et al. (2023).

3. EVOLUTIONARY COMPUTING FOR CONTROLLER DESIGN

Designing control strategies for multi-agent robotic systems with realistic dynamics and coordination constraints remains a significant challenge. *Evolutionary computing* (EC) offers a powerful alternative by enabling the automatic synthesis of control policies through optimization. Among EC methods, *genetic programming* (GP) is particularly suited for evolving interpretable symbolic expressions (see Brameier and Banzhaf (2007); Koza (1994)). In GP, each individual is a tree mapping inputs (Fig. 1) (e.g., position or velocity errors) to control outputs, constructed from predefined primitives (e.g., sum, multiply) and terminals (e.g., constants, random values). A population of such trees is evaluated using task-specific fitness functions, and evolved via selection, crossover, and mutation.

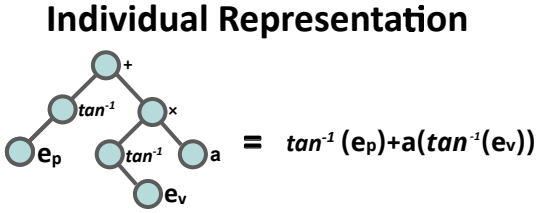

Fig. 1. Example tree representation of an Individual

In swarm and aerial robotics, Rezk et al. (2015) applied EC to tasks such as obstacle avoidance, flocking, and coordination Sperati et al. (2008), especially in scenarios where analytical models are hard to define or manual tuning is intractable. Recent studies, such as Jones et al. (2018); Nikolos et al. (2007), have explored EC for evolving behavior trees, reactive policies, and symbolic controllers in both simulation and hardware.

In this work, we use GP to evolve artificial potential field controllers for quadcopters. The evolved expressions operate directly on vector and scalar inputs (e.g., position and velocity errors), enabling the discovery of novel, modular strategies for attraction, repulsion, and synchronization, promoting interpretability and reuse across tasks.

4. METHODOLOGY

We adopt a hierarchical evolutionary approach using Genetic Programming (GP) to design control functions for multi-agent quadrotor coordination. The control strategy is developed in three sequential stages: attraction, synchronization, and formation. Each stage builds upon the evolved controllers from the previous one, promoting modular and interpretable behaviors.

4.1 Stage 1: Attraction Function Evolution

In the first stage, we evolve an attraction function to drive a single agent toward a fixed goal position. The evolved function $\mathbf{f}_{\text{att}}$ maps:

$$\mathbf{f}_{\text{att}} : (\mathbf{e}_p, \mathbf{e}_v, \mathbf{v}_{dp}, \|\mathbf{e}_p\|) \to \mathbb{R}^3$$

where $\mathbf{e}_p = \mathbf{p}_i - \mathbf{p}_g$ is the position error between the agent and the target, $\mathbf{e}_v = \mathbf{v}_i - \mathbf{v}_g$ is the velocity error, $\mathbf{v}_{dp}$ is the desired velocity derivative, and $\|\mathbf{e}_p\|$ is the distance to the goal.

The fitness function is the average Euclidean position error over the trajectory:

 https://doi.org/10.58571/CNCA.AMCA.2025.069

$$\text{Fitness}_{\text{att}} = \frac{1}{T}\sum_{t=1}^{T} \|\mathbf{p}(t) - \mathbf{p}_{\text{goal}}(t)\|$$

The best individual is saved and used in subsequent stages as a fixed baseline controller.

4.2 Stage 2: Synchronization Function Evolution

In the second stage, we evolve a synchronization function $\mathbf{f}_{\text{sync}}$ to minimize inter-agent velocity discrepancies. The previously evolved attraction function is used without modification. The input features to the synchronization function remain the same.

The agents are initialized at different positions, and the synchronization controller is tasked with reducing their velocity differences while converging to the same goal. The fitness function is:

$$\text{Fitness}_{\text{sync}} = \frac{1}{T}\sum_{t=1}^{T} \|\mathbf{v}_i(t) - \mathbf{v}_{\text{goal}}(t)\|$$

where $\mathbf{v}_i(t)$ is the velocity the agent at time t and $\mathbf{v}_{\text{goal}}$ is the target velocity.

4.3 Stage 3: Formation Control via Repulsion

The final stage introduces a repulsion component $\mathbf{f}_{\text{rep}}$ to maintain a target inter-agent distance and avoid collisions. The full control law becomes:

$$\mathbf{u}_i = \mathbf{f}_{\text{att}} + \mathbf{f}_{\text{sync}} + \mathbf{f}_{\text{rep}}$$

The repulsion function $\mathbf{f}_{\text{rep}}$ depends on:

- Relative vector to neighbor $\mathbf{ep}_{\text{ij}} = \mathbf{p}_i - \mathbf{p}_j$.
- Distance to neighbors $\|\mathbf{ep}_{\text{ij}}\|$.
- Desired distance between agents $\mathbf{d}$.
- Relative velocity to neighbor $\mathbf{ev}_{\text{ij}} = \mathbf{v}_i - \mathbf{v}_j$.

$$\mathbf{f}_{\text{rep}} : (\mathbf{ep}_{\text{ij}}, \|\mathbf{ep}_{\text{ij}}\|, \mathbf{ev}_{\text{ij}}, \mathbf{d}) \to \mathbb{R}^3$$

The fitness function penalizes deviation from a desired inter-agent spacing d and proximity violations:

$$\text{Fitness}_{\text{form}} = \frac{1}{T}\sum_{t}\sum_{i<j} \left(\|\Delta_{ij}\| - d\right)^2 + \mathbf{k}_{\text{r}}\,\mathbb{I}_{\|\Delta_{ij}\|<d_{\min}}$$

where $\Delta_{ij} = \mathbf{p}_i - \mathbf{p}_j$ denotes the relative position between agents i and j. Here, $\mathbf{k}_{\text{r}}$ is a penalty coefficient that amplifies the cost when the distance between agents i and j falls below a minimum threshold $d_{\min}$, thus discouraging collisions. $\mathbb{I}$ denotes the indicator function (equal to 1 if the condition holds and 0 otherwise).

4.4 Evolutionary Setup

All controllers are encoded as typed GP trees using the DEAP library. The primitive set includes vector operations, scalar-vector arithmetic, norms, nonlinearities (e.g., `tanh`, `log`, `exp`), and ephemeral constants. See Table 1

The evolution parameters are:

Table 1. Primitive set used in Genetic Programming

Type	**Operations**
Vector-Vector	$+$, $-$, $\times$, $\div$
Vector-Scalar	`vec` $\pm$ `scalar`, `vec` $\cdot$ `scalar`, `vec`/`scalar`
Scalar-Vector	`scalar` $\pm$ `vec`, `scalar` $\cdot$ `vec`, `scalar`/`vec`
Vector Unary	$\log(\boldsymbol{x})$, $\exp(\boldsymbol{x})$, $\boldsymbol{x}^2$, $\sqrt{\boldsymbol{x}}$
Scalar Unary	$\log(x)$, $\exp(x)$, x^2, $\sqrt{x}$, $\sin(x)$, $\cos(x)$, $\tanh(x)$
Vector Norms	$\|\boldsymbol{x}\|$, $\boldsymbol{x}/\|\boldsymbol{x}\|$
Scalar Ops	$+$, $-$, $\times$, $\div$, $\|x\|$, $-x$
Constants	1.0, -1.0, 0.5, -0.5, π
Ephemerals	Random vector, random scalar

- Population size: 100
- Generations: 100
- Selection: Tournament (size 3)
- Crossover: One-point
- Mutation: Uniform on full trees
- Tree depth limit: 12

Agent trajectories are visualized in 3D, enabling qualitative assessment of navigation, synchronization, and formation behavior.

5. RESULTS

Agents simulation is based on a 6-DOF rigid-body dynamic model, including translational and rotational dynamics (Mellinger and Kumar, 2011), which is widely used in quadrotor control literature. Each simulation runs for 5 seconds with a timestep of 10 ms. Fitness values are computed per stage, and the best individual is stored after evolution. Individuals are expression trees that combine vector and scalar operations to define control laws; they are evolved to minimize a fitness function based on navigation performance. The simulation begins with the drones positioned at the following coordinates:

$$\mathbf{p}_1(0) = \begin{bmatrix}0.5\\0.0\\0.0\end{bmatrix}, \quad \mathbf{p}_2(0) = \begin{bmatrix}-0.250\\0.433\\0.0\end{bmatrix}, \quad \mathbf{p}_3(0) = \begin{bmatrix}-0.250\\-0.433\\0.0\end{bmatrix}$$

These initial positions were chosen to ensure that the drones are evenly distributed around the origin $(0, 0, 0)$, forming a symmetric configuration. Constants k_a, k_s, and k_r were chosen to improve the performance of the desired behavior.

5.1 Evolved Attraction Controller

The attraction component was evolved using Genetic Programming (GP) to drive each agent toward a static target located at $(0, 0, 0.5)$ in 3D space.

Table 2 shows some of the Evolved Attraction Forces and Mean Position Error (MPE) comparison with the attraction APF mentioned in Section 2. Lower Mean Positional Error, the closer the agents reach and remain in the target position.

The evaluation criterion was the mean positional error accumulated over the simulation execution. A value above 0.3 indicates, as shown in Fig. 2 and Fig. 3, that the agents reach the target position, but its movement doest end at that point. The agents passes the target, and then they are pulled back. The evolved controller demonstrated

 https://doi.org/10.58571/CNCA.AMCA.2025.069

Table 2. Attraction controller comparison (lower is better).

Controller	**MPE**
Evolved Individual 1	
$\log(\tanh(\pi)) \cdot \mathbf{e}_p$	0.4200
Evolved Individual 2	
$\tanh(\sin(\|\mathbf{e}_p\|)) \cdot [(\sin(\|\mathbf{e}_p\|) - \|\mathbf{e}_p\|) \cdot \mathbf{e}_p]$	0.4546
Evolved Individual 3	
$-0.01143 \cdot \|\mathbf{e}_p\| \cdot \mathbf{e}_p$	0.3912
Quadratic ($k_a = 0.2$)	0.4746
Conic ($k_a = 0.2$)	0.5469
Logarithmic ($k_a = 0.2$)	0.4934

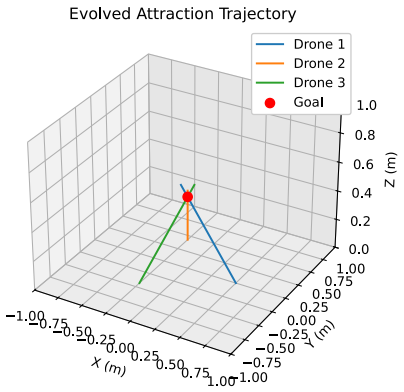

Fig. 2. Evolved Individual 1 Attraction Trajectory

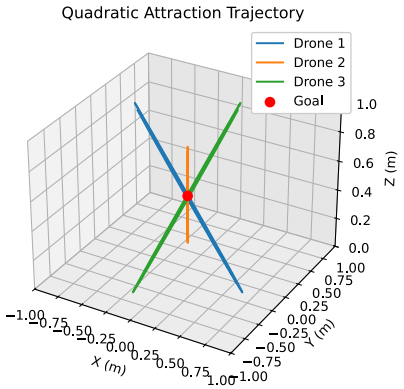

Fig. 3. Quadratic Attraction Trajectory

superior performance when compared to classical potential field formulations. The evolved controller exhibited a more direct and stable convergence to the target, with reduced overshoot and trajectory oscillations, as evident from both the fitness metrics and trajectory.

5.2 Evolved Synchronization Term

To promote coherent group behavior, a synchronization term was evolved that operates on the relative error (e_p), velocity error (e_v), and desired velocity (v_{dp}). When combined with the evolved or benchmark attraction term, this component significantly reduced inter-agent velocity discrepancies. The fitness function was based on the average magnitude of velocity errors over time (Mean Velocity Error, MVE). The resulting controllers produced trajectories in which agents maintained temporal coherence during movement, facilitating coordinated navigation toward the target location as shown in Fig. 4.

Table 3. Synchronization controller comparison (lower is better).

Controller	**MVE**
Evolved Individual 1	
$-0.5 \cdot (4\mathbf{e}_p + \mathbf{e}_v)$	0.0290
Evolved Individual 2	
$\frac{-0.5}{\sqrt{\sqrt{\mathbf{e}_p^2 - 0.5}}} \mathbf{v}_{dp} + \mathbf{e}_v$	0.0358
Evolved Individual 3	
$-0.5 \cdot (\mathbf{e}_v + 4\mathbf{e}_p)$	0.0379
Linear Damping ($k_s = 0.2$)	0.0425
Exponential Decay ($k_s = 0.2$)	0.0455
Saturated Alignment ($k_s = 0.2$)	0.0471

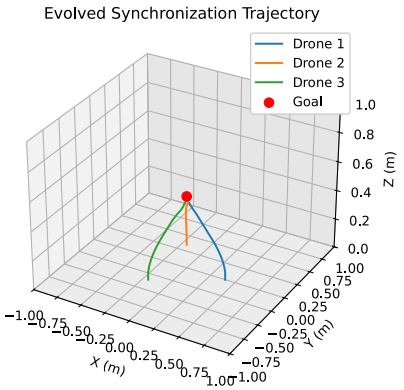

Fig. 4. Evolved Individual 1 Synchronization Trajectory

5.3 Formation Maintenance via Evolved Repulsion

Formation control was achieved by evolving a repulsion term to regulate inter-agent distances. The fitness function penalized both excessive proximity (collision risk at 0.2 m) and deviation from a desired target formation radius (set at 0.3 m). The repulsion controllers were trained using GP with inputs including pairwise distance vectors and velocity information, trying to maintain the Mean Formation Error (MFE) as close to 0 as possible. Simulation results presented in Table 4 and in Fig. 5 show that the evolved term maintained agents within the desired formation envelope while enabling progression toward the global objective. A negative MFE means the drone has moved inside the desired formation radius, being closer to the target than intended. The emergent behavior exhibited consistent spatial distribution with no observed collisions throughout the trials.

5.4 Integrated Multi-Agent Behavior (Trajectory)

To assess generalization and robustness, the evolved attraction, synchronization, and repulsion controllers were

 https://doi.org/10.58571/CNCA.AMCA.2025.069

Table 4. Formation control (repulsion) performance (lower is better).

Controller	MFE
Evolved Individual 1 $-\mathbf{e}_v \cdot \tanh(\mathbf{v}_{dp})$	-0.0260
Evolved Individual 2 $\left(\frac{\mathbf{e}_v}{\text{direction}}\right) \cdot \left(\mathbf{v}_{dp} - 0.5\right) \cdot \mathbf{v}_{dp}$	-0.0870
Evolved Individual 3 $\mathbf{e}_v \cdot \sin(-0.3)$	-0.0271
Planar distance-based repulsion ($k_r = 0.2$)	0.1789
Linear ($k_r = 0.1$)	-0.1619
Inverse Distance ($k_r = 0.1$)	-0.1574

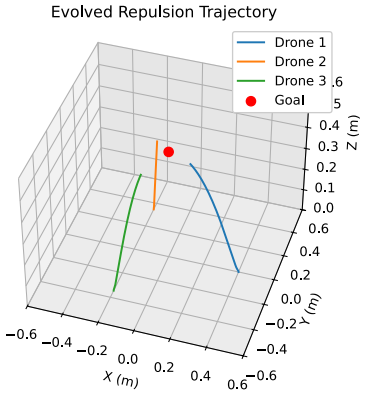

Fig. 5. Evolved Individual 1 Repulsion Trajectory

evaluated in a scenario involving a moving target. Specifically, the target followed a circular trajectory in the horizontal plane, centered at $(0, 0, 0.5)$ with a radius of 0.5 meters and angular velocity $\omega = \frac{2\pi}{T}$, completing one revolution over the course of the simulation.

Under this dynamic reference, agents were initialized at random positions and tasked with tracking the moving target while maintaining inter-agent spacing and velocity synchronization. The individuals used in this evaluation, as well as the classical APF functions selected for comparison, are shown in Table 5.

$$\mathbf{u}_i = \mathbf{f}_{\text{att}} + \mathbf{f}_{\text{rep}} + \mathbf{f}_{\text{sync}}, \tag{2}$$

Table 5. Function Selection.

Evolved Controller	APF Controller
$\mathbf{f}_{\text{att}}$: $\log(\tanh(\pi)) \cdot \mathbf{e}_p$	$\mathbf{f}_{\text{att}}$: Quadratic Attraction
$\mathbf{f}_{\text{sync}}$: $-0.5 \cdot (4\mathbf{e}_p + \mathbf{e}_v)$	$\mathbf{f}_{\text{sync}}$: Linear Damping
$\mathbf{f}_{\text{rep}}$:$-\mathbf{e}_v \cdot \tanh(\mathbf{v}_{dp})$	$\mathbf{f}_{\text{rep}}$: Planar distance-based repulsion

Table 6. Trajectory performance comparison (lower is better).

Controller	MPE	MFE
Evolved Controller	0.0245	-0.0260
APF Controller	0.0814	0.087

The results shown in Table 6 demonstrated that the controllers generalized beyond static conditions: all agents successfully followed the circular trajectory with minimal positional lag and velocity mismatch. The repulsion term effectively preserved inter-agent distances, preventing collisions despite continuous reorientation and acceleration.

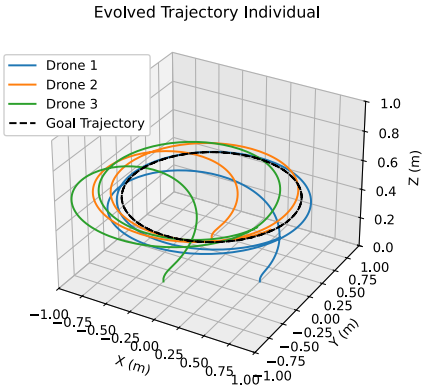

Fig. 6. Evolved Trajectory Individual

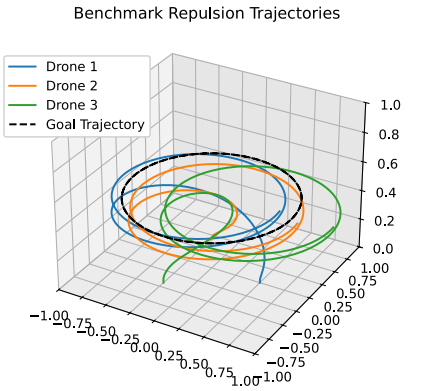

Fig. 7. APF Trajectory Individual

The quadcopters demonstrated coordinated and coherent motion along the trajectory of the moving target, maintaining a loose yet consistent circular formation around it as shown in Fig 6. The MPE and MFE across multiple runs remained low, indicating reliable performance under dynamic conditions. These findings highlight the capacity of the evolved modular potential field components to generalize effectively to time-varying objectives, underscoring their suitability for decentralized control in multi-agent systems operating in non-static environments.

6. CONCLUSION

This paper presented an evolutionary framework for synthesizing artificial potential field controllers for swarm coordination in aerial robotics. Using strongly typed genetic programming, we evolved interpretable, vector-valued control laws for attraction, repulsion, and synchronization, which operate directly on kinematic variables such as position and velocity errors. These controllers were evaluated in a realistic simulation of quadrotor dynamics, demonstrating smooth convergence, inter-agent coordination, and robustness to initial conditions.

In future work, we aim to validate the evolved controllers on real hardware using the Crazyflie platform. We plan to transfer the evolved expressions to onboard control code and deploy them in an indoor motion capture environment.

 https://doi.org/10.58571/CNCA.AMCA.2025.069

This will allow us to assess real-world performance under sensing noise, latency, and thrust limitations. Additional directions include evolving behaviors in cluttered spaces, adapting to dynamic obstacles, and combining our approach with learning-based perception modules.

ACKNOWLEDGEMENTS

This work was supported by the Secretariat of Science, Humanities, Technology and Innovation (SECIHTI)

REFERENCES

(2017). Linear vs nonlinear mpc for trajectory tracking applied to rotary wing micro aerial vehicles. *IFAC-PapersOnLine*, 50(1), 3463–3469. doi: https://doi.org/10.1016/j.ifacol.2017.08.849. 20th IFAC World Congress.

Arnold, R.D., Yamaguchi, H., and Tanaka, T. (2018). Search and rescue with autonomous flying robots through behavior-based cooperative intelligence. *Journal of International Humanitarian Action*, 3(1), 1–18.

Besada, E., de la Torre, L., de la Cruz, J., and Andres-Toro, B. (2010). Evolutionary trajectory planner for multiple uavs in realistic scenarios. *Robotics, IEEE Transactions on*, 26, 619 – 634. doi: 10.1109/TRO.2010.2048610.

Bouček, Z., Flídr, M., and Straka, O. (2024). Trajectory tracking for uavs: An interpolating control approach. *arXiv preprint arXiv:2407.01095*.

Brameier, M. and Banzhaf, W. (2007). *Linear Genetic Programming*. Springer.

Debie, E., Kasmarik, K., and Garratt, M. (2023). Swarm robotics: A survey from a multi-tasking perspective. *ACM Comput. Surv.*, 56(2). doi:10.1145/3611652. URL `https://doi.org/10.1145/3611652`.

Fujimori, A., Kubota, H., Shibata, N., and Tezuka, Y. (2014). Leader-follower formation control with obstacle avoidance using sonar-equipped mobile robots. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 228, 303–315. doi:10.1177/0959651813517682.

Handayani, A., Nurmaini, S., and Yani, I. (2015). Brief review on formation control of swarm robot. doi: 10.11591/eecsi.2.804.

Horváth, D., Gazda, J., Šlapak, E., Maksymyuk, T., and Dohler, M. (2021). Evolutionary coverage optimization for a self-organizing uav-based wireless communication system. *IEEE Access*, 9, 145066–145082.

Huang, Y., Tang, J., and Lao, S. (2019). Uav group formation collision avoidance method based on second-order consensus algorithm and improved artificial potential field. *Symmetry*, 11(9). doi:10.3390/sym11091162. URL `https://www.mdpi.com/2073-8994/11/9/1162`.

Jones, S., Studley, M., Hauert, S., and Winfield, A. (2018). Evolving behaviour trees for swarm robotics. In *Distributed autonomous robotic systems: The 13th international symposium*, 487–501. Springer.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1), 90–98.

Klančar, G., Zdešar, A., and Krishnan, M. (2022). Robot navigation based on potential field and gradient obtained by bilinear interpolation and a grid-based search. *Sensors*, 22(9). doi:10.3390/s22093295. URL `https://www.mdpi.com/1424-8220/22/9/3295`.

Koza, J.R. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4, 87–112.

li Su, J. and Wang, H. (2021). An improved adaptive differential evolution algorithm for single unmanned aerial vehicle multitasking. *Defence Technology*, 17(6), 1967–1975. doi:https://doi.org/10.1016/j.dt.2021.07.008.

Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, 2520–2525. doi:10.1109/ICRA.2011.5980409.

Meza-Sánchez, M., Clemente, E., Rodríguez-Liñán, M., and Olague, G. (2019). Synthetic-analytic behavior-based control framework: Constraining velocity in tracking for nonholonomic wheeled mobile robots. *Information Sciences*, 501, 436–459. doi: https://doi.org/10.1016/j.ins.2019.06.025.

Neupane, A. and Goodrich, M.A. (2019). Learning swarm behaviors using grammatical evolution and behavior trees. In *IJCAI*, 513–520.

Nikolos, I.K., Zografos, E.S., and Brintaki, A.N. (2007). Uav path planning using evolutionary algorithms. In *Innovations in intelligent machines-1*, 77–111. Springer.

Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3), 401–420.

Qin, H., Shao, S., Wang, T., Yu, X., Jiang, Y., and Cao, Z. (2023). Review of autonomous path planning algorithms for mobile robots. *Drones*, 7(3), 211.

Ren, W. and Beard, R.W. (2005). Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, 50(5), 655–661.

Rezk, N.M., Alkabani, Y., Bedour, H., and Hammad, S. (2015). Swarm robotics obstacle avoidance: A progressive minimal criteria novelty search-based approach. In *Progress in Artificial Intelligence: 17th Portuguese Conference on Artificial Intelligence, EPIA 2015, Coimbra, Portugal, September 8-11, 2015. Proceedings 17*, 493–498. Springer.

Rossi, F., Bandyopadhyay, S., Wolf, M., and Pavone, M. (2018). Review of multi-agent algorithms for collective behavior: a structural taxonomy. *arXiv preprint arXiv:1803.05464*.

Selvam, P.K., Raja, G., Rajagopal, V., Dev, K., and Knorr, S. (2021). Collision-free path planning for uavs using efficient artificial potential field algorithm. In *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 1–5. IEEE.

Sfeir, J., Saad, M., and Saliah-Hassane, H. (2011). An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment. In *2011 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, 208–213.

Sperati, V., Trianni, V., and Nolfi, S. (2008). Evolving coordinated group behaviours through maximisation of mean mutual information. *Swarm Intelligence*, 2, 73–95.

Zhao, Z. et al. (2023). Enhanced trajectory tracking for quadrotors: disturbance observer-based control. *PeerJ Computer Science*, 9, e1861.

 https://doi.org/10.58571/CNCA.AMCA.2025.069