COMPUTATIONAL PROBLEMS OF ELECTRICAL ENGINEERING
Vol. 15, No. 1, 2025

USE OF SWARM INTELLIGENCE IN UNMANNED VEHICLES

Anastasiia Bodnar, Ihor Rabiichuk, Pavlo Serdyuk, Andrii Fechan
Lviv Polytechnic National University, Lviv, Ukraine
anastasiia.bodnar.pz.2022@lpnu.ua, *ihor.o.rabiichuk@lpnu.ua*,
pavlo.v.serdiuk@lpnu.ua, *andrii.v.fechann@lpnu.ua*

https://doi.org/10.23939/jcpee2025.01.007

Abstract: This article explores the use of swarm intelligence algorithms in unmanned vehicles (UVs), focuses on their main advantages for improving the efficiency and productivity of systems. Unmanned vehicles, which can operate autonomously or under remote control, play a significant role in such areas as surveillance, search and rescue, agriculture and military operations. The main focus of the article is on algorithms such as ant colony optimisation (ACO), artificial bee colony (ABC), particle swarm optimization (PSO), glow-worm swarm optimization (GSO), firefly algorithm (FA), bat algorithm (BA), grey wolf optimizer (GWO), and whale optimization algorithm (WOA). Each of these algorithms is discussed in detail, particularly their core principles, specific applications in UVs, and their levels of effectiveness in different environments. Each algorithm has been examined to highlight its operational strengths and its limitations, such as computational demands and environmental suitability. This paper discusses the algorithms in terms of managing critical functions of UVs, such as resource allocation and multi-agent coordination, which are essential for complex mission scenarios. Particular attention is paid to the adaptability of each algorithm, especially in unpredictable or hostile environments, where rapid recalibration of UV behaviour is necessary for mission success. By analysing each algorithm capacity to adjust the UV to new data in real-time, the article highlights their potential to optimize UV performance and reliability in challenging contexts. Special attention is given to collaborative task management in swarm intelligence, emphasizing its ability to enhance unmanned aerial vehicle (UAV) group coordination and decision-making for efficient operation in complex and dynamic scenarios. In general, the article provides deep analysis of swarm intelligence algorithms, and the information that will help choose the most effective algorithm to help solve specific tasks using different types of UVs. Future research will focus on improving the scalability, adaptability, and integration of these algorithms with latest technologies in order to enhance their effectiveness in solving complex UV missions. In addition, a comparative table of the main characteristics of the algorithms was created and a review of similar studies comparing swarm algorithms was made.

Key words: Multi-agent systems, optimization algorithms, task coordination, adaptive systems, dynamic scenarios.

1. Introduction

Unmanned vehicles (UVs) are a type of vehicle that can be controlled remotely or programmed to perform a task autonomously. It is hard to deny that UVs have rapidly gained popularity and now play a significant role in all areas of human life. They can be used for surveillance, search and rescue, agriculture and forestry, as well as military purposes. Swarm intelligence is one of the most significant areas in the study of algorithms for UVs. Swarm intelligence is a field that aims to build fully distributed decentralised systems in which the overall functionality of the system arises from the interaction of individual agents with each other and the environment. To effectively perform the tasks, appropriate algorithms are used, such as ant colony optimisation (ACO), artificial bee colony (ABC), particle swarm optimisation (PSO), glow-worm swarm optimisation (GSO), firefly algorithm (FA), bat-inspired algorithm (BA), grey wolf optimiser (GWO), whale optimisation algorithm (WOA). They are used to solve problems related to route planning, target search, resource allocation, and coordination of UV groups. However, despite a large number of studies in this area, the question of the appropriate and effective use of algorithms for different tasks and types of UVs still remains relevant.

The main objective of this study is to review and analyse existing swarm intelligence algorithms and their application to improve the performance and efficiency of UAVs. This article analyses the advantages and disadvantages of algorithms in different usage conditions and provides recommendations for the effective application of a particular algorithm for a given task and UAV, for example, in cases of military operations, environmental research, or disaster relief.

The object of research is the processes of applying and implementing swarm intelligence in unmanned vehicles.

The subject of research are algorithms and methods of swarm intelligence applied to different types of unmanned vehicles.

The purpose of the work is to study and analyse the use of swarm intelligence in unmanned vehicles to improve their performance and efficiency.

To achieve this purpose, the following *main research objectives* are identified:

- analyse existing swarm intelligence algorithms;
- evaluate the performance of these algorithms;
- identify the advantages and disadvantages.

Materials and methods of research. In the work the following materials and methods were used: modern swarm intelligence algorithms, such as Particle Swarm Optimisation (PSO), Ant Colony Optimisation (ACO) and Artificial Bee Colony (ABC), glow-worm swarm optimisation (GSO), firefly algorithm (FA), bat-inspired algorithm (BA), grey wolf optimiser (GWO), whale optimisation algorithm (WOA); methods for analysing the performance of algorithms in unmanned vehicles (UVs); methods of data processing and estimation to determine the effectiveness of algorithms in different conditions.

Analysis of recent research and publications. One of the global challenges in the application of swarm intelligence algorithms for unmanned vehicles is the problem of adaptation to specific environmental conditions and tasks. Many scientific papers have been published on this topic, the article *Swarm Intelligence: Concepts, Models, and Applications* being one of them, which addresses the global problem of using swarm intelligence in UVs, in particular, improving the efficiency of algorithms for different tasks and types of UVs. The authors analysed the existing approaches and methods, highlighted their pros and cons in different applications. They emphasise that the effectiveness of the algorithm selection depends on the specific conditions and type of the task being performed by the UV [1]. One of the main aspects highlighted by the authors of *A Review of Swarm Robotics Tasks* is the need for a more detailed study and adaptation of swarm intelligence algorithms to specific conditions of use, which will ensure more efficient operation of UAVs in various fields, such as search and rescue, surveillance, and other tasks [2]. In the article *Swarm Based Optimisation Algorithms For Task Allocation In Multi-Robot Systems: A Comprehensive Review*, the authors describe swarm intelligence algorithms in detail, consider their underlying principles, their application in various fields, advantages, disadvantages, and highlight the problem of task allocation in multi-robot systems, including unmanned vehicle systems. According to the authors, this work can lead to understanding the swarm optimization algorithms which will enhance the possibility of applying multi-robot systems to solve real-world problems with increasing levels of complexity [3]. As for more specific examples in the article *Motion Planning of UAV Swarm: Recent Challenges and Approaches*, the authors consider the challenges associated with the use of UAV in different tasks and suggest improved algorithms to enhance the efficiency of the tasks. The authors have explored methods and models of swarm planning: control, route planning, architecture, communication, monitoring and tracking as well as security issues. Furthermore, the authors highlight that the efficiency of algorithms may also range significantly from one specific environment and task to another, which raises a question about further improvement in their performance [4]. In recent years, autonomous underwater vehicles also have made significant progress, especially in the fields of oceanography, military and underwater research. The authors of the article *Path Planning for Autonomous Underwater Vehicles* say that route planning is really important in allowing AUVs to operate efficiently: helping them navigate properly to avoid getting stuck and use less energy. The authors proposed addressing this issue by merging two algorithms in order to enhance the computational efficiency [5]. The article *Review of Multiple Unmanned Surface Vessels Collaborative Search and Hunting Based on Swarm Intelligence* highlights the challenges in coordinating multiple unmanned surface vessels for tasks like cooperative search and hunting. The authors discuss the advantages and limitations of swarm intelligence algorithms, emphasizing their ability to enhance collaboration and improve task efficiency. They propose optimization methods to address existing challenges and suggest future research directions, such as adapting algorithms to dynamic environments and improving fault tolerance in Multiple Unmanned Surface Vessel (MUSV) systems [6]. Many approaches are used today to improve the efficiency of algorithms, as shown above, scientists also combine algorithms to achieve the highest performance. Thus, a comprehensive analysis of the literature on the subject of the study suggests that increasing the efficiency of adaptation and application of swarm intelligence algorithms for different types of unmanned vehicles and usage conditions remains an urgent problem. Developers suggested some methods for solving these problems, however further research and improvement of existing methods are required to ensure the effective outcome of operations of unmanned vehicles in various applications.

2. Research results and their discussion

Ant colony optimization (ACO). The ant colony optimization algorithm is widely used in UVs for solving complex route planning and optimization problems. The algorithm mimics how ants find food using pheromone traces. The ant chooses the next step based on how many pheromones other ants have left along the path (i.e., the most visited places and thus considered to be the optimal solution) and how heuristically attractive the path is (e.g., short). This approach helps the algorithm to avoid getting stuck on early stages and helps to find new paths using pheromone evaporation. The key formula is the next step (path) probability formula [16]:

$$P_{ij} = \frac{(\tau_{ij})^{\alpha} (\eta_{ij})^{\beta}}{\sum_{k \in N_i} (\tau_{ik})^{\alpha} (\eta_{ik})^{\beta}}, \quad (1)$$

where τ_{ij} is the pheromone level and $\eta_{ij} = \frac{1}{d_{ij}}$ the heuristic visibility. Considering two alternative paths $i \to j$ and $i \to k$ with $\tau_{ij} = 4$, $\tau_{ik} = 2$, $\alpha = 1$, $\beta = 2$, and distances $d_{ij} = 10$, $d_{ik} = 5$, as a result, we get $P_{ij} = 0.33$ and $P_{ik} = 0.67$. Each local ratio is scaled linearly, so despite slow initial convergence, computational cost remains moderate and rarely grows exponentially. However, when environments shift quickly, updating τ_{ij} can lag, making real-time responses low or moderate at best.

The main advantage of the ACO is its ability to distribute computation and avoid premature convergence. The ACO is often used in tasks that require finding globally optimal paths. One of the biggest advantages is fast convergence in the later stages, as well as a quick approach to the optimal or near-optimal solution, especially in the final stages of the search. Another key advantage is memorability, when pheromone traces help to 'remember' previous decisions to influence future ones. However, tuning parameters like α, β, and pheromone decay is crucial; poor values may cause early convergence or inefficient exploration. Multiple UAVs share pheromone updates, allowing them to benefit from other partial solutions, though concurrent updates can sometimes conflict [17, 20]. As for scalability, each denominator demands *M* numeric products and additions thus overhead is O(*M*). Larger swarms may strain runtime, but basic expansions still work well. The ACO can handle multiple objectives by combining them into a single weighted heuristic η_{ij}. For example, distance plus energy e_{ij} can be combined as $\eta_{ij} = \frac{\omega_1}{d_{ij}} + \frac{\omega_2}{e_{ij}}$, where ω is their weights. This approach is feasible but remains simpler than full multi-objective optimization since it reduces multiple goals to one scalar. The ACO performs best in static environments but can be adapted for dynamic ones. One of the key drawbacks is its slow convergence during the initial iterations due to the reliance on accumulated pheromones, which delays path discovery. Additionally, fixed parameters, especially the pheromone evaporation rate, can result in premature convergence or prolonged exploration. To improve performance, a dynamic evaporation rate is proposed: a higher rate during early iterations to encourage diverse exploration, followed by a gradual decrease to stabilize the convergence around promising solutions. Also, local pheromone resets in areas with rapidly changing data can prevent UAVs from relying on outdated route data. These adjustments improve responsiveness and path precision in semi-dynamic environments. Overall, the ACO is best for moderate-speed route optimization with partially changing constraints, performing well for tasks like stable obstacle avoidance or multi-target navigation, but less ideal where real-time reactivity and massive scalability dominate [3, 8, 12].

Artificial bee colony (ABC). An artificial bee colony algorithm, inspired by the honeybees, is widely used for route planning for Unmanned Combat Aerial Vehicles (UCAVs) due to its effective balance between exploration and exploitation. It models the behaviour of three types of bees: employed, onlooker and scout. Local search is performed by employed and onlooker bees, improving on solutions already found. Employed bees generate new solutions by mutating existing good ones, then onlooker bees select the best solutions for improvement. This allows the algorithm to quickly converge to optimal or near-optimal solutions. Meanwhile, scout bees perform a global search function, randomly looking for new solutions. This helps to avoid local minimums, which is important for finding optimal routes. However, the local exploitation of the ABC is less efficient than its global search, which slowly converges to the optimal solution when fast decisions are needed. ABC main update formula is [18]:

$$v_i = x_i + \phi_{i,k}(x_i - x_k), \quad (2)$$

where x_i is the current path, x_k a random neighbour, and $\phi_{i,k}$ is a random number between −1 and 1. This method helps explore possible routes. However, if the "limit" (the number of unsuccessful tries before a bee abandons a solution) is too high, the algorithm reacts slowly to sudden changes, making real-time performance only low to moderate. The ABC is easy to implement, requiring just a few parameters. However, performance is sensitive to the "limit" and colony balance, so some tuning is needed, because incorrect settings of these parameters can lead to suboptimal performance or premature convergence. In the ABC, onlooker bees choose paths based on fitness scores [18]:

$$pi = \frac{f_i}{\Sigma f_i}. \quad (3)$$

For example, with scores [0.2, 0.3, 0.5, 0.8, 1.2], the best path (1.2) gets a 40 % chance of being picked. This selection uses simple math, just additions and divisions, so the algorithm does about *O(N)* operations. That means computational cost grows steadily, keeping ABC efficient for mid-sized problems while still focusing on better routes. The ABC has linear complexity *O(N)*, making it efficient for mid-sized UAV groups. For larger swarms, performance may drop unless optimized. Collaboration in the ABC is indirect, bees share solution quality through fitness values, not pheromones. Onlooker bees focus on strong paths, while scouts explore new ones. This decentralized system supports flexible coordination, but with less global awareness than the ACO or the PSO [17, 20]. The ABC handles multiple goals by combining them into one using weighted sums which allow easy comparison. However, it lacks advanced methods like Pareto fronts, limiting flexibility in complex trade-offs. A key drawback of the Artificial Bee Colony (ABC) algorithm is its slow local convergence caused by strong global search and weak local refinement. The fixed "limit" parameter for abandoning poor solutions also reduces responsiveness in dynamic environments. To improve adaptability, a dynamic limit based on environmental changes or convergence speed is proposed. Lowering the limit in unstable conditions enables quicker solution updates. Additionally, enhancing local search using gradient-based or Lévy-flight-inspired movements can boost refinement. These changes aim to improve real-time ABC responsiveness while preserving its strengths in decentralized coordination for dynamic UAV route optimization. Overall, the ABC is ideal for pre-flight planning, threat avoidance, and target allocation, but not suited for fast-changing, real-time UAV control [3, 7, 13].

Particle Swarm Optimization (PSO). A Particle swarm optimization algorithm is inspired by the behaviour of birds forming flocks to search for food. It uses the collective behaviour of the particles to get global optimisation with an iterative method, where particles adjust to their trajectory after considering individual experience and interaction with others, similar to how birds will not collide with others in flight. The PSO has been widely used in obstacle avoidance and trajectory optimization tasks for unmanned underwater vehicles. The algorithm finds the most efficient and safe route in an underwater environment, considering constraints such as obstacles, depth and the need to minimise power consumption. By considering the UUV trajectories as particles in a swarm, the PSO dynamically re-optimizes these trajectories based on the data received from vehicles. This helps the UUV to follow the correct path and avoid stalls even with sudden change of trajectory. The PSO updates particles using [19]:

$$v_i(t+1) = \omega v_i(t) + c_1 r_1 (p_i - x_i) + c_2 r_2 (g - x_i), \quad (4)$$

$$x_i(t+1) = x_i(t) + v_i(t+1). \quad (5)$$

Here, x_i is the current drone route, p_i its personal best, and g the swarm best. The values ω, c_1, and c_2 control momentum, personal memory, and group influence. In dynamic settings, relying too much on past bests can slow adaptation, so improved PSO variants adjust these values or add randomness for faster response. Each PSO iteration performs three operations per particle: velocity update *O(D)*, position update *O(D)*, and fitness evaluation *O(1)* or *O(D)*. Total cost is $N = (2D + \mathit{fitness\,complexity}) \approx O(ND)$, scaling steadily without combinatorial growth, making the PSO efficient for real-time UV tasks. The total effort of the PSO scales linearly: *T* = *O(ND)*, where *N* = number of particles, *D* = dimensionality. Doubling either particles or dimensions doubles workload. However, since the PSO does not require population-wide interaction, it remains scalable in distributed UV applications. The standard PSO handles multiple objectives by scalarizing them. This approach simplifies optimization, but can miss Pareto-optimal trade-offs. At the initial stage, the PSO has a high convergence rate, but at later stages it can slow down, which may require additional improvements to the algorithm to increase its efficiency. One of the main disadvantages of the PSO is that it can get stuck in local optima, leading to inefficient task performance. It is easy to implement the PSO with just three main parameters and no mutation, crossover, or pheromones. Defaults often work well, though tuning improves results. Initialization and boundaries are easier than in many other algorithms. Swarm coordination in the PSO relies on the shared global best g, guiding each agent update. Though agents do not communicate directly, their actions stay aligned, enabling coordinated UV behaviour without central control [17, 20]. A key limitation of the standard Particle Swarm Optimization (PSO) is its tendency to get trapped in local optima, especially in later stages, reducing adaptability in dynamic UUV environments. Its reliance on historical best positions hinders responsiveness to rapid changes. To address this, a re-randomization mechanism for stagnating particles, those without recent improvement, can be introduced. Additionally, the use of adaptive inertia weighting, which decreases over time, encourages early exploration and later convergence. These enhancements improve the resilience, adaptability, and real-time performance of the PSO while preserving its simplicity and speed, making it more effective for UV coordination and continuous path re-optimization. The PSO is suited to real-time routing, multi-UV coordination, low-resource systems, and swarm control. However, it is less effective with rapidly changing goals or strict discrete constraints [1, 8, 15].

Glow-Worm Algorithm (GSO). The algorithm is inspired by the behaviour of fireflies, which emit light using a chemical called luciferin. They use it to attract prey and coordinate their swarming movements. This behaviour is the main principle of the GSO algorithm, making it useful in tasks requiring path optimization. This approach balances exploring new solutions and exploiting current ones using the ability of fireflies to dynamically adapt their search radius. The GSO updates each agent's luciferin (signal strength) and position using [21]:

$$l_i(t+1) = (1-p)l_i(t) + \gamma J(x_i(t)), \quad (6)$$

$$x_i(t+1) = x_i(t) + s \cdot \frac{x_j(t) - x_i(t)}{\left\| x_j(t) - x_i(t) \right\|}, \quad (7)$$

where l_i is luciferin, p is the decay constant, γ is luciferin gain, and x_j is a neighbour with a stronger signal. The GSO forms multiple swarms for flexible adaptation, but its responsiveness depends on decay and sensing range, which may limit speed in fast-changing conditions. Thus, the GSO differs from other algorithms, because it does not focus on finding a single global solution, but looks for multiple solutions with varying values of the objective function. The algorithm divides the swarm into groups, each converging to different local optima, allowing the UAV to explore multiple viable paths simultaneously. This decentralized behaviour enables parallel search and coordination, making it ideal for multi-objective or multi-region UAV tasks [20]. The GSO helps the UAV avoid premature convergence to suboptimal solutions, a common problem in dynamic environments. By directing each firefly (representing the UAV) to brighter, more promising areas in the search space, the GSO decreases the risk of falling into local optimum. In each iteration, the position of the UAV is modified and the focus is modulated by the luciferin level described as the attractive potential of a location in the search space. Each GSO iteration involves luciferin update *O(1)*, neighbour checking *O(N)*, and movement update *O(D)*. Since each of the *N* agents compares itself to all others, total cost becomes $O(ND+N^2)$. This $O(N^2)$ neighbour search is a GSO main bottleneck. To address this, it is proposed to introduce a fixed-radius or grid-based neighbourhood filtering strategy. This would reduce comparison overhead to *O(kN)*, making the algorithm significantly more scalable for large UAV swarms. Additionally, GSO responsiveness in fast-changing environments is limited by static luciferin decay settings. A potential solution is to implement a dynamic decay adjustment, where the decay rate increases in rapidly changing zones to encourage faster response and new path discovery. The GSO relies on tuning four key parameters: step size, luciferin decay, gain, and neighbourhood range. Though being simple in design, its performance depends heavily on these settings: too small the radius limits collaboration, too large the convergence slowdown. Proper tuning is essential but manageable. The GSO uses a fitness function, which can incorporate weighted criteria. This scalarization supports multi-objective optimization in a basic form. The GSO does not natively support Pareto fronts, but it handles multi-criteria routing well with custom scoring functions. In general, GSO excels in distributed surveillance, search and rescue, and adaptive navigation. It is less suited for time-critical updates or very large swarms without neighbourhood filtering [8, 9, 14].

Firefly algorithm (FA). The firefly algorithm is inspired by fireflies and their bioluminescent behaviour, where flashing signals indicate attractive positions, guiding the search for optimal solutions. The FA is widely used for optimizing flight paths in complex and dynamic battlefield conditions. In UAV applications, the FA starts with a random set of solutions representing potential flight paths, maintaining diversity in the solution space and avoiding premature convergence to local optima. It allows UAVs to efficiently avoid threats and minimise fuel consumption while exploring and converging on optimal routes. The FA updates positions based on the formula [22]:

$$x_i(t+1) = x_i(t) + \beta_0 e^{-\gamma r_{ij}^2}\left(x_j(t) - x_i(t)\right) + \alpha\epsilon_i(t), \quad (8)$$

where x_i is the current solution, x_j is a more attractive firefly, r_{ij} is the distance between them, and $\alpha\epsilon_i(t)$ adds randomness. This balances exploration and exploitation, by guiding each'firefly' towards brighter, promising areas in the search space, ensuring UAVs find efficient routes. However, in fast-changing environments, the standard FA may converge slowly or follow outdated paths unless enhanced with adaptive steps or random walk. For *N* fireflies in a *D*-dimensional space, each firefly compares itself to all others ($N-1$ interactions), leading to a complexity of $O(N^2D)$. Each interaction includes a distance calculation *O(D)*, brightness comparison *O(1)*, and position update *O(D)*. This quadratic scaling makes the FA less efficient for large swarms but still manageable in mid-sized UAV groups. The FA scales poorly as each firefly compares with all others, causing comparisons to grow rapidly with a swarm size. To address this, introducing a fixed interaction radius or neighbourhood filtering would reduce comparisons to nearby agents only, improving scalability to *O(kN)* and allowing the FA to scale to larger UAV teams. Another issue is slow convergence in dynamic environments, where outdated solutions may mislead the swarm. A possible solution is to incorporate an adaptive randomization factor that increases when environmental changes are detected, allowing the algorithm to re-diversify and respond more effectively to sudden threats. The FA uses just a few parameters: light absorption γ, attractiveness β_0, and randomization factor α. The algorithm is easy to implement, but performance is sensitive to tuning. Poor settings can cause early convergence or weak exploration, especially in complex and high-dimensional UAV tasks. The FA enables implicit coordination as fireflies move toward brighter neighbours, guiding UAVs to promising areas. Without global memory, swarms may cluster around local optima unless randomness is added [17]. The FA typically uses scalar fitness values. Multi-objective routing is handled by combining factors into one score, while being simple, this limits adaptability. The FA does not support Pareto front maintenance unless extended with specialized Multi-Objective Optimization (MOO) techniques. The FA is extremely flexible, fitting well into the conditions under which many UAVs must operate where threats can change. If a new threat is detected, the FA recalibrates the flight path by recalculating the brightness of fireflies to avoid the threat. Due to its reliable and flexible planning, UAVs maintain mission effectiveness with minimal risk. In general, the firefly algorithm is a powerful tool for UAV flight planning. The FA is best for offline planning, smooth trajectories, and moderately dynamic tasks. It is less suited to real-time replanning or large UAV swarms [10, 11, 14].

Bat-inspired Algorithm (BA). The bat algorithm is an advanced swarm intelligence technique inspired by the echolocation behaviour of bats. The main principle of the algorithm is that bats navigate by emitting sound pulses and analyse the echoes returning from objects, building a 3D map of their surroundings. The algorithm copes with navigation in three-dimensional space, where the main goal is to determine the accident-free, shorter and safer flight path. This makes the BA an invaluable tool for UAVs that need to cross difficult terrain while avoiding threats. The BA updates positions with a velocity-frequency model [23]:

$$v_i^t = v_i^{t-1} + \left(x_i^t - x^*\right) f_i \,, \tag{9}$$

$$x_i^t = x_i^{t-1} + v_i^t \,, \tag{10}$$

where f_i is frequency, and x^* is the global best. The algorithm explores the search space where potential solutions are represented by bats which move towards more promising regions. As they approach better solution, the algorithm adjusts its exploration, reducing the volume and frequency of the bats' pulses. This way the BA focuses on improving solutions while ensuring that UAVs find optimal flight paths with a reduced collision risk and minimal exposure to threats. However, standard BA may lag in fast-changing situations without adaptive control mechanisms. Each bat updates frequency, velocity, position, and, optionally, loudness, with total cost per iteration *O(ND)*. This makes the BA lighter than the FA or the ACO and suitable to real-time or embedded UAV tasks. The BA does not require pairwise comparisons. Each agent updates using only its own state and the global best. This yields linear complexity, scaling well even with large swarms. The BA uses a few parameters, hence the algorithm is simple to implement, but performance depends heavily on balancing exploration and exploitation. Coordination is implicit. Bats adjust their path using the global best solution, which enables alignment toward a shared goal without direct communication. Though less interactive than the PSO, this mechanism supports emergent coordination in UAV groups [17, 20]. The standard BA also combines multiple criteria into a single scalar. The BA is ideal for fuel-efficient routing, smooth trajectories, and energy-aware target tracking. It is less suited to fast responses under high uncertainty without enhancements. The BA is highly effective in global optimization, such as UAV trajectory planning, where it navigates through dynamic terrains and adapts to changing conditions with minimal computational cost. However, it can sometimes struggle with local search, causing it to get stuck in local optima. To overcome this, a local intensification phase can be introduced (such as a refined random walk triggered when swarm diversity drops), allowing UAVs to escape local optima without sacrificing global convergence. Additionally, the performance of the BA heavily depends on parameter settings like a pulse rate and frequency. A promising solution is to apply adaptive parameter tuning, where these values adjust dynamically based on convergence speed or environmental volatility. This would improve the algorithm responsiveness in rapidly changing conditions of the UAV mission. With powerful global search capabilities, the BA is widely used in tasks requiring navigation through complex and dynamic environments [7, 11, 15].

Grey Wolf Optimizer (GWO). The grey wolf optimization algorithm models the hierarchical structure, behaviour, and hunting mechanisms of grey wolves. One of the key features of the GWO is its ability to model collective search, where agents (wolves) play different roles depending on their position in the pack. α wolves direct the search and select the optimal route, β and δ wolves refine it, while ω wolves follow the collective strategy. This way the GWO provides position updates that do not focus only on the position of one leader, which helps to avoid premature convergence to a single local optimum [24]:

$$\vec{X}\left(t+1\right) = \frac{\vec{X}_\alpha + \vec{X}_\beta + \vec{X}_\delta}{3} \,. \tag{11}$$

In dynamic UAV tasks, the GWO adapts well due to this leader-based approach, allowing agents to react to changing conditions collaboratively. Studies have shown the GWO outperforms several metaheuristics in terms of path cost and convergence when navigating dynamic or multi-target environments. In UAVs, this behaviour means efficient search and optimisation of flight paths in three-dimensional space with constraints. The algorithm ability to balance between exploration and exploitation makes it well suited to finding optimal or near-optimal paths for UAVs, allowing them to cover large areas efficiently while avoiding threats. Avoiding premature convergence is a key benefit of the GWO, allowing UAVs to find globally optimal routes in complex environments. However, in large-scale, optimization problems the GWO can suffer from, are premature convergence, leading to suboptimal solutions. Additionally, GWO risks falling into local optima in complicated and multi-dimensional environments, causing suboptimal waypoint selection. To address this, a diversity control mechanism can be introduced–such as periodically reinitializing a portion of the omega wolves or injecting controlled noise into position updates to maintain exploration and prevent early stagnation. Additionally, the performance of the GWO in complex environments can be improved by adopting a dynamic role adaptation, where the leadership structure (α, β, δ) is periodically reassessed based on recent fitness improvement rather than static rankings. This would promote adaptability and robustness during long or dynamic UAV missions. For N agents in D dimensions each agent evaluates fitness $O(1)$, updates positions based on 3 best agents → $O(D)$, total per iteration $T = O(ND)$. No pairwise comparison is required, making the GWO more efficient than the FA or the ACO in computation. Because the GWO relies only on global bests and not agent-to-agent comparison, the algorithm scales linearly with population size. This makes the GWO suitable for large UAV fleets and swarm control with minimal bottlenecks. The GWO has minimal parameters, and its update mechanism is simple. However, some sensitivity to population size and the balance between exploration and exploitation exists, especially in complex environments [17, 20]. A GWO hierarchical model enables decentralized coordination, with agents guided by alpha, beta, and delta wolves, being ideal for multi-UAV systems. It avoids scalarization by tracking multiple top solutions, preserving swarm diversity and supporting trade-offs across objectives like distance, energy, and risk. The GWO remains scalable and suitable for large groups where coordination and coordinated search are needed. It is easy to see how this would be ideal for military applications where the terrain cannot always be assured to be safe [6, 11, 14].

Whale Optimization Algorithm (WOA). The Whale Optimization Algorithm is inspired by the bubble net hunting strategy of humpback whales. It mimics the social behaviour and hunting mechanisms of whales, in particular their method of catching prey using a spiral bubble net. The encirclement stage allows gradual approach to the best solution. The WOA is effectively used to optimise path tracking and navigation for unmanned vehicles. The WOA identifies the best (with least errors) navigation paths, ensuring safe and efficient navigation for autonomous vehicles in challenging environments, key formula for it being:

$$\vec{X}(t+1)=\begin{cases}\vec{X}_*(t)-A\cdot D,\ p<0{,}5\\ \vec{D}'\cdot e^{bl}\cdot\cos(2\pi l)+\vec{X}_*(t),\ p\geq 0{,}5\end{cases}. \tag{12}$$

Here, $\vec{X}_*(t)$ is the best solution so far, and the random number p switches between spiral and circular paths. This mechanism helps the WOA adapt by either exploring new regions or exploiting known good areas. However, like most population-based algorithms, standard responsiveness of the WOA to sudden environmental changes is limited without adaptation techniques. For each whale in a population of size N and dimension D, the position update requires best-solution distance $O(D)$ and randomized movement $O(D)$. Total cost per iteration is $T = O(ND)$. This is efficient and suitable for onboard UAV systems, especially when real-time constraints are moderate. The WOA does not perform pairwise comparisons and only requires knowledge of the global best, so it scales linearly with swarm size. For large UAV groups, it remains computationally practical. All whales adjust their paths based on the current best solution, which drives collective behaviour. While not explicitly collaborative like the ACO, the WOA enables implicit coordination, making it useful for decentralized UAV mission planning [20]. The WOA typically scalarizes multiple objectives into a single fitness value, this allowing basic trade-offs (e.g., between distance, energy, and risk). An important advantage of the WOA is its adaptability to different environmental conditions. The WOA requires minimal parameters (spiral coefficient b, coefficients A and C, and random factors). It is simple to implement, but performance varies with the balance between exploration (spiral movement) and exploitation (shrinking encirclement). However, algorithm balances exploration and exploitation, by adaptively changing trajectories in the search space, helping reduce overfitting to local optima. Another benefit is that this optimization converges to a better solution in fewer iteration compared with other algorithms, making it well-suited for real-time applications, which requires quick responses. However, performance of the algorithm largely depends on selection of initial parameters and optimization with the WOA can become complex when tasks are very dynamic or large in scale. Also, the WOA sometimes falls into local optima, especially in complex and multidimensional search spaces. Thus, a key limitation of the WOA is its limited responsiveness in highly dynamic environments, where the reliance on a single best solution can cause the swarm to stagnate or overfit to outdated paths. To address this, a multi-leader memory mechanism can be introduced, where a small archive of recent best solutions guides exploration. This would diversify the search and enable quicker adaptation to environmental changes. Additionally, to reduce the risk of local optima entrapment, incorporating a randomized reinitialization for stagnant agents could periodically introduce diversity, helping UAVs escape misleading trajectories and improving overall robustness in complex path planning tasks. The WOA suits to energy-efficient routing, semi-dynamic planning, and smooth trajectories. Overall, its adaptability and efficiency make it a valuable tool in the development of autonomous UAV systems [11, 14].

For better understanding and comparison, the properties of ACO, ABC, PSO, GSO, FA, BA, GWO, WOA are presented in Table.

ACO, ABC, PSO, GSO, FA, BA, GWO, WOA comparison

	ACO	ABC	PSO	GSO	FA	BA	GWO	WOA
Performance in Dynamic Environments	Low-Moderate	Low-Moderate	Low-Moderate	Moderate-High	Moderate-High	Moderate-High	Moderate-High	Low-Moderate
Computational Efficiency	Moderate	Moderate	Moderate	Moderate	Moderate	High	High	High
Scalability	Low	Moderate	Moderate	Moderate	Moderate–High	High	High	High
Ease of Implementation and Fine-Tuning	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate
Optimal Use Cases	Route Optimization	Pre-flight & mid-scale planning	Real-time UAV coordination	Multi-region UAV search,	Threat avoidance, mission planning	Dynamic tracking with limited resources	Multi-coordination and threat-aware routing	Energy-efficient routing with low reactivity
Collaborative Task Management	Moderate	Moderate	Moderate–High	Moderate–High	Moderate–High	Moderate	High	Moderate
Ability to Handle Multiple Objectives	Limited (weighted scalar heuristic, no Pareto support)	Limited (scalarized objectives, no Pareto front)	Limited (scalarization, true multi-objective needs MOPSO)	Limited (basic scalarization, no Pareto support)	Limited (scalar scores, lacks Pareto tracking)	Limited (scalar fitness, multi-objective support limited)	High (tracks multiple top solutions without scalarization)	Limited (scalar objective, lacks Pareto front unless extended)

Discussion of research results. This paper examines different systems and algorithms associated with swarm intelligence techniques that are used in unmanned vehicle (UV) systems. Results have suggested that each of these algorithms has its own different areas of strength and weaknesses that affect their efficiency under various operating conditions. The ant colony optimization (ACO) algorithm has shown high efficiency in routing and path optimization tasks, especially in conditions where high accuracy and efficiency are required. On the other hand, it has been noticed that the ACO has limitations in the late stages of searching, which may slow down the convergence process and make this method less suitable in areas where fast response is required [12]. The artificial bee colony (ABC) algorithm has demonstrated high adaptability to dynamic environments and the ability to effectively avoid local minima, which is critical for military operations and rescue missions [13]. However, experiments have shown that this algorithm has some difficulties with fast convergence at local levels, which may require additional parameter optimization to improve its efficiency. The particle swarm optimization (PSO) algorithm has proven to be one of the best tools for optimising trajectories in complex environments such as underwater vehicles (UUVs). One of its strong features is the ability to quickly adapt to changes and ensure safe navigation. However, the downside of the PSO is that it quite often results in premature convergence to local optima, which can reduce the overall efficiency of the algorithm in complex and multidimensional spaces [14, 15]. The glow-worm swarm optimization (GSO) and the firefly algorithm (FA) are highly effective in coordinating the exploration of several alternative optimal routes simultaneously by UVs [14]. This makes them useful tools for solving tasks that require high flexibility and the ability to adapt to new conditions. Despite the previous statement, the quality of the results from these algorithms depends on the set of the parameters, which may be problematic under challenging operating conditions. The bat algorithm (BA) has shown significant performance on

3D navigation tasks, but its local search may not be so effective, sometimes leading to getting stuck in local optimum [15]. The grey wolf optimizer (GWO) and whale optimization algorithm (WOA) have demonstrated high efficiency in providing stable and safe navigation in complex and dynamic environments [14]. However, they can also experience the problem of premature convergence of the optimization process, which limits their use in circumstances that require global optimization. In conclusion, some difficulties arise in the course of optimising the navigation systems of the unmanned vehicles (UVs) in complex and dynamic environments. While many of the swarm intelligence algorithms have undergone some improvements, the challenge of global optimization without settling for suboptimal or local optima still persists. This is a serious concern because it impacts the ability of the unmanned vehicles to perform with high accuracy and reliability, especially when there is a need to respond to quick and unpredictable environmental changes. Another important problem is the adjustment of algorithm parameters that significantly influences their efficiency. If the tuning is incorrect, then the solution is not optimal and will degrade the performance of the system as well as the overall success of the mission. This problem becomes even more relevant when a task is to be performed in real time, when the speed of decision making as well as the accuracy in doing this is very crucial. Therefore, the main problem is the enhancement of swarm intelligence algorithms so that they can be used in dynamic and complex environments and development of the adaptive parameter control techniques to improve reliability and stability of unmanned systems. So, basing on the results of the work performed, it is possible to formulate the following scientific novelty and practical significance of the research results.

Scientific novelty of the obtained research results is a comprehensive analysis and comparison of various swarm intelligence algorithms used in unmanned systems.

Practical significance of the research results lies in the possibility of optimising navigation systems for UVs, which allows for increasing their efficiency and reliability when performing tasks in difficult conditions. In particular, the results of the study can be used to improve control and navigation systems, which will ensure the safe operation of autonomous vehicles in various environments.

Conclusions

The paper presents an in-depth review of the performance and potential of different swarm intelligence, that is, driven algorithms and their incorporation in unmanned vehicles to address the problems of route optimization and navigation in a complex and dynamic environment. Various algorithms were investigated, including Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), Particle Swarm Optimization (PSO), Glow-Worm Swarm Optimization (GSO), Firefly Algorithm (FA), Bat Algorithm (BA), Grey Wolf Optimizer (GWO), and Whale Optimization Algorithm (WOA). The results show that these algorithms demonstrate significant potential for tasks such as route planning, resource allocation, and real-time adaptation to changes in the environment, but there are challenges, in particular, in their adapting to dynamic environments. Each algorithm has its own unique advantages, but a common problem is the risk of premature convergence to local optima, which limits their effectiveness in global optimization. Also, it has been established that the adjustment of the parameters of the algorithms is very important for the quality of the work done, especially in real time.

This convergence allows more robust decision-making processes, improved adaptability, and effective coordination in complex tasks. In order to improve their scalability and adaptability for use in the UVs, further research should focus on developing hybrid approaches that take advantage of multiple algorithms. Such enhancements are fundamental, taking into account the need to ensure operational efficiency in unmanned vehicles that have to operate under optimal but rather undesirable conditions while performing complicated tasks.

References

[1] A. Hazem and J. Glasgow, *Swarm intelligence: concepts, models and applications*, School of Computing, Queen's University Technical Report. 2012. https://doi.org/10.13140/2.1.1320.2568

[2] L. Bayindir, "A review of swarm robotics tasks", *Neurocomputing,* No. 172*,* pp. 292–321, 2016. https://doi.org/10.1016/j.neucom.2015.05.116

[3] V. Dabass, et al., "Swarm-based optimization algorithms for task allocation in multi-robot systems: A comprehensive review", *Educational Administration: Theory and Practice,* vol. 30, No. 4, pp. 2689–2695, 2024. https://doi.org/10.17762/ijritcc.v11i9.10484

[4] M. Iqbal, Z. Ali, R., Khan, and M. Shafiq, "Motion planning of UAV swarm: Recent challenges and approaches", *IntechOpen,* 2022. https://doi.org/10.5772/intechopen.106270

[5] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, and D. Lane, "Path planning for autonomous underwater vehicles", *Proc. IEEE Transactions on Robotics,* vol. 23, No. 2, pp. 331–341, 2007. https://doi.org/10.1109/TRO.2007.895057

[6] G. Wu, T. Xu, Y. Sun, and J. Zhang, "Review of multiple unmanned surface vessels collaborative

search and hunting based on swarm intelligence”, *International Journal of Advanced Robotic Systems*, vol. 19, No. 2, 2022. https://doi.org/10.1177/17298806221091885

[7] T. Islam, M. Islam, and M. Ruhin, “An analysis of foraging and echolocation behavior of swarm intelligence algorithms in optimization: ACO, BCO, and BA”, *International Journal of Intelligence Science,* vol. 8, No. 1, pp. 1–27, 2018. https://doi.org/10.4236/ijis.2018.81001

[8] J. Luo, Y. Tian, and Z. Wang, “Research on unmanned aerial vehicle path planning”, *Drones,* vol. 8, No. 2, p. 51, 2024. https://doi.org/10.3390/drones8020051

[9] Z. Tang and Y. Zhou, “A glowworm swarm optimization algorithm for uninhabited combat air vehicle path planning”, *Journal of Intelligent Systems,* vol. 24, No. 1, pp. 69–83. 2015. https://doi.org/10.1515/jisys-2013-0066

[10] G.-G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, “A modified firefly algorithm for UCAV path planning”, in *Proc. 5th International Conference on Natural Computation (ICNC)*, pp. 264–268, 2011.

[11] A. Chakraborty and A. Kar, “Swarm intelligence: A review of algorithms. In A. Abraham and V. Krzhizhanovskaya (Eds.)”, *Innovations in Swarm Intelligence and Optimization.* Springer, pp. 19–52, 2017. https://doi.org/10.1007/978-3-319-50920-4_19

[12] M. Mavrovouniotis, C. Li, and S. Yang. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. *Swarm and Evolutionary Computation,* No. 33, pp. 1–17, 2017. https://doi.org/10.1016/j.swevo.2016.12.005

[13] S. Sabet, M. Shokouhifar, and F. Farokhi, “A comparison between swarm intelligence algorithms for routing problems”, *Electrical & Computer Engineering: An International Journal,* No. 1, 5, pp. 17–33, 2016. https://doi.org/10.14810/ecij.2016.5102

[14] H. Zhu, Y. Wang, Z. Ma, and X. Li, “A comparative study of swarm intelligence algorithms for UCAV path-planning problems”, *Mathematics,* vol. 9, No. 2, p. 171, 2021. https://doi.org/10.3390/math9020171

[15] J. Fan, M. Hu, X. Chu, and D. Yang, “A comparison analysis of swarm intelligence algorithms for robot swarm learning”, in *Proc. Winter Simulation Conference (WSC)*, pp. 3042–3053, 2017. https://doi.org/10.1109/WSC.2017.8248025

[16] Marco Dorigo and Thomas. Stützle, *Ant Colony Optimization: Overview and Recent Advances*, 2010. https://doi.org/10.1007/978-1-4419-1665-5_8

[17] S. Darvishpoor, A. Darvishpour, M.Escarcega, and M. Hassanalian, “Nature-Inspired Algorithms from Oceans to Space: A Comprehensive Review of Heuristic and Meta-Heuristic Optimization Algorithms and Their Potential Applications in Drones”, *Drones*, No. 7, p. 427, 2023. https://doi.org/10.3390/drones7070427

[18] Dharmender Kumar and S.K. Jarial, “A Review on Artificial Bee Colony Algorithms and Their Applications to Data Clustering”, *Cybernetics and Information Technologies*, No. 17. pp. 3–28, 2017. https://doi.org/10.1515/cait-2017-0027

[19] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh and S. Mirjalili, “Particle Swarm Optimization: A Comprehensive Survey”, in *Proc. IEEE Access*, vol. 10, pp. 10031–10061, 2022. https://doi.org/10.1109/ACCESS.2022.3142859

[20] Agrawal Sameer, K. Patle Bhumeshwar and Sanap Sudarshan, “A systematic review on metaheuristic approaches for autonomous path planning of unmanned aerial vehicles”, *Drone Systems and Applications*, No. 12, pp. 1–28, 2024. https://doi.org/10.1139/dsa-2023-0093

[21] P. S. Ramesh, P. Srivani, M. Mahdal, L. Sivaranjani, S. Abidin, S. Kagi, and M. Elangovan, “Contextual Cluster-Based Glow-Worm Swarm Optimization (GSO) Coupled Wireless Sensor Networks for Smart Cities”, *Sensors*, No. 23, p. 6639, 2023. https://doi.org/10.3390/s23146639

[22] Himanshu Verma, *A Systematic Review on Firefly Algorithm: Past, Present, and Future*, 2020. https://doi.org/10.36227/techrxiv.12122748

[23] Xin-She Yang, A *New Metaheuristic Bat-Inspired Algorithm*, p. 284, 2010. https://doi.org/10.1007/978-3-642-12538-6_6

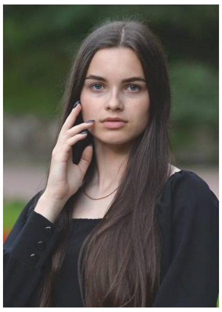
[24] Emmanuel Dada, Stephen Joseph, David Oyewola, Alaba Fadele, Haruna Chiroma, and Shafi’I Abdulhamid, “Application of Grey Wolf Optimization Algorithm: Recent Trends, Issues, and Possible Horizons”, *Gazi University Journal of Science*. No. 35. pp. 485–504, 2022. https://doi.org/10.35378/gujs.820885

[25] M. Nadimi-Shahraki, H. Zamani, Varzaneh Asghari, Z. et al. “A Systematic Review of the Whale Optimization Algorithm: Theoretical Foundation, Improvements, and Hybridizations”, *Arch Computat Methods Eng*, No. 30, pp. 4113–4159, 2023. https://doi.org/10.1007/s11831-023-09928-7.

ВИКОРИСТАННЯ РОЙОВОГО ІНТЕЛЕКТУ В БЕЗПІЛОТНИХ АПАРАТАХ

Анастасія Боднар, Ігор Рабійчук, Павло Сердюк, Андрій Фечан

Досліджено використання алгоритмів ройового інтелекту в безпілотних апаратах (БПА), акцентовано увагу на їх значних перевагах для підвищення ефективності та продуктивності цих систем. Безпілотні апарати, які можуть функціонувати автономно або під дистанційним керуванням, відіграють ключову роль у таких сферах, як спостереження, пошуково-рятувальні операції, сільське господарство та військові дії. Основний фокус статті зосереджено на таких алгоритмах, як оптимізація мурашиних колоній (ACO), штучна бджолина колонія (ABC), оптимізація рою частинок (PSO), оптимізація рою світлячків (GSO), алгоритм світлячка (FA), алгоритм кажана (BA), оптимізація сірого вовка (GWO) та алгоритм оптимізації китів (WOA). Ґрунтовно проаналізовано принципи роботи кожного алгоритму, їхнє застосування у БПА, а також оцінено їхню ефективність у динамічних умовах. Також простежено ключові переваги кожного алгоритму та їхні обмеження, такі як потреба в обчислювальних ресурсах і відповідність певному середовищу. Алгоритми розглянуто з точки зору управління критично важливими функціями БПА, такими як розподіл ресурсів і координація дій у багатоагентних системах, що є важливим для виконання складних місій. Особливу увагу зосереджено на адаптивності кожного алгоритму, особливо в умовах непередбачуваного та складного середовища, де швидка зміна поведінки БПА може визначати успіх місії. Окрім цього, зосереджено увагу на здатності кожного алгоритму адаптуватися до нової інформації в режимі реального часу, що відкриває перспективи для підвищення продуктивності та надійності БПА у складних умовах. Окремий акцент зроблено на координації завдань у ройовому інтелекті, підкреслюючи його здатність покращувати групову взаємодію безпілотних апаратів (БПА) та ухвалення рішень для ефективної роботи у складних і динамічних умовах. Запропоновано глибокий аналіз алгоритмів ройового інтелекту та надано рекомендації щодо вибору найбільш ефективного підходу залежно від специфіки завдання і типу БПА. Крім того, створено порівняльну таблицю ключових властивостей алгоритмів та зроблено огляд аналогічних досліджень, що порівнюють ройові алгоритми. Майбутні дослідження будуть зосереджені на вдосконаленні масштабованості, адаптивності та інтеграції цих алгоритмів із новітніми технологіями для розв’язання складних завдань у місіях БПА.



Anastasiia Bodnar – Student of the Software Engineering program at the Institute of Computer Science and Information Technologies, Lviv Polytechnic National University, Ukraine. Her scientific activity includes swarm intelligence algorithms, unmanned vehicle systems, optimization methods. She pays great attention to their use in autonomous multi-agent coordination and distributed computing systems.

Ihor Rabiichuk is a PhD student at the Institute of Computer Science and Information Technologies, Lviv Polytechnic National University, Ukraine.

His research focuses on swarm intelligence algorithms in dynamic environments, their application in adaptive and distributed systems, as systems, optimization methods well as the use of cellular automata for modeling and optimization processes.

Pavlo Serdyuk is an Associate professor of the Software Engineering Department at the Lviv Polytechnic National University, Ukraine. His academic and research interests include software architecture, artificial intelligence applications, optimization methods, and the development of intelligent systems for healthcare, autonomous agents, and complex multi-component platforms.

Andriy Fechan D. Sc., Prof. Head of Department of Electronics and Information Technology of Lviv Polytechnic National University. Research interests cover optoelectronic devices based on organic materials and computer simulations of anisotropic liquids by molecular dynamics methods-complex multi-component platforms.

Research results are published in more than 200 scientific papers and 25 patents by Ukraine.

Received: 30.07.2025, Accepted: 25.08.2025
ORCID 0009-0006-0891-9865 (Bodnar A.)
ORCID 0009-0002-0881-0406 (Rabiichuk I.)
ORCID 0000-0002-2677-3170 (Serdyuk P.)
ORCID 0000-0001-9970-5497 (Fechan A.)