Section: Information Technology, Cyber Security and Computer Engineering

DOI 10.70286/EOSS-02.02.2026.003.68-72

ЗАДАЧА СКЛАДАННЯ МАРШРУТУ БПЛА ДЛЯ ОБСТЕЖЕННЯ ЗОН ОБ'ЄКТІВ

Жданова Олена
к.т.н., доцент
Попенко Володимир
к.т.н., доцент
Заскалета Богдан
здобувач вищої освіти
Лєпєєв Володимир
здобувач вищої освіти
Кафедра інформаційних систем та технологій
Національний технічний університет України
«Київський політехнічний інститут», Україна

Безпілотні літальні апарати (БПЛА) набули широкого застосування не лише у військовій сфері, а й у сільському господарстві та логістиці різних галузей. Ефективність їх використання безпосередньо визначається раціональним управлінням ресурсами БПЛА. До найбільш критичних ресурсів належать дальність польоту, а також тісно пов'язані з нею тривалість польоту та допустима маса корисного навантаження.

Змістовна постановка задачі. Нехай одному БПЛА треба обстежити кілька об'єктів, тобто пролетіти на відстані від об'єкту достатньо близько. Маємо множину об'єктів $J = 1,2,\dots,n$, які необхідно обстежити за допомогою БПЛА, (x_j, y_j) - координати об'єкту $j(j = 1,2,\dots,n)$. БПЛА без перезарядки може пролетіти L од. відстані. Відомі також координати двох точок базування БПЛА: A та B з координатами (x_A, y_A) та (x_B, y_B) відповідно. БПЛА стартує з бази A та приземляється на базі B. Необхідно скласти план польоту БПЛА, за якого він, перелітаючи з A в B, зміг би по дорозі обстежити максимальну кількість об'єктів. Об'єкт вважається обстеженим, якщо БПЛА наблизиться до об'єкту хоча б на r од. відстані (тобто, якщо БПЛА пролетить над зоною радіусом r центром якого є цей об'єкт).

Приклад розміщення баз A та B та $n = 4$ об'єктів, а також один із припустимих розв'язків – маршрут руху БПЛА – наведено на рис. 1.

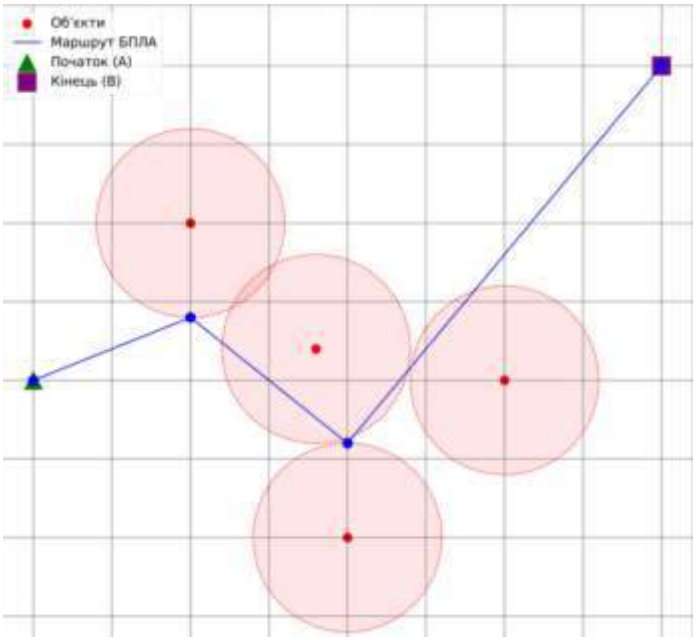

Рисунок 1. Приклад маршруту БПЛА обстеження об’єктів для задачі з $n = 4$

Математична постановка задачі. Планування маршруту БПЛА зводиться до побудови кусково-лінійної (ламаної) траєкторії, що з’єднує задані точки старту та фінішу, а також (за потреби) проходить через додаткові точки – точки розвороту, і задовольняє обмеження на максимальну дальність польоту. Оскільки обстеження об’єктів може здійснюватися з будь-якої точки маршруту, виникає допоміжна задача — визначити, які саме об’єкти покриває кожний відрізок ламаної з урахуванням допустимого радіуса розпізнавання. Наприклад, на рис. 1 другий відрізок маршруту дозволяє обстежити три об’єкти.

Вхідні дані: n – кількість об'єктів для обстеження; $j = (1, 2, \dots, n)$ – об'єкти які потрібно обстежити; (x_j, y_j) – координати об'єкта $j = (1, 2, \dots, n)$; $A(x_A, y_A)$ – стартова точка маршруту; $B(x_B, y_B)$ – кінцева точка маршруту; L - максимальна відстань, яку може пролетіти БПЛА без перезарядки; r – радіус (дальність) розпізнавання об’єкта.

Шукані величини. Необхідно знайти: m – кількість точок маршруту (точок розвороту) з обстеження об’єктів БПЛА; $O = (o_1, o_2, \dots, o_m)$ – послідовність точок маршруту, де $(o_2, \dots, o_{m-1})$ – точки розвороту; (x_{o_i}, y_{o_i}) - координати точок маршруту $o_i, i = \overline{1, \dots, m}$; $P(O) = (o_i, o_{i+1}) \mid i = \overline{1,2, \dots, m-1}) = (p_1, p_2, \dots, p_{m-1})$ – множина відрізків маршруту (ребер ламаної).

Обмеження:

$$o_1 = A,\ o_m = B,$$

$$\sum_{i=1}^{m-1} l_i \le L,$$

де $l_i = \sqrt{(x_{o_i} - x_{o_{i+1}})^2 + (y_{o_i} - y_{o_{i+1}})^2}, i = \overline{1,2, \dots, m-1}$ – довжина відрізка між послідовними точками маршруту.

Цільова функція – максимізація кількості обстежених об'єктів на складеному маршруті БПЛА:

$$z = \left| \bigcup_{i=1}^{m-1} K([p_i]) \right| \to \max,$$

де $K([p_i]) = \{j, 1 \le j \le n$ такі, що $d(j, [p_i]) \le r\}$ - множина об'єктів, для яких відстань до відрізка $[p_i]$ не перевищує відстані розпізнавання r.

Для вирішення задачі були розроблені і порівнювались два алгоритми: жадібний та метаевристичний.

Жадібний алгоритм є простим евристичним методом оптимізації, який виконується одноразово та приймає рішення на основі локально оптимальних виборів. Алгоритм будує маршрут без гарантії глобальної оптимальності, проте враховує реальні обмеження на дальність польоту та умови обстеження об'єктів. На кожному кроці алгоритм обирає найближчий необстежений об'єкт, який може бути відвіданий з поточної позиції за умови дотримання обмеження на максимальну довжину маршруту (відстань, яку БПЛА може пролетіти без перезарядки). Після вибору наступної точки розвороту алгоритм перевіряє, чи може БПЛА з нової позиції досягти кінцевої точки прямим перельотом (без відвідування додаткових об'єктів). Якщо після обстеження обраного об'єкта залишкової дальності недостатньо для досягнення кінцевої точки, такий об'єкт відкидається, а БПЛА прямує з поточної точки безпосередньо до фінішу. На завершальному етапі, коли прийнято рішення летіти до кінцевої точки, для кожного необстеженого об'єкта обчислюється відстань до відрізка прямого польоту (від поточної позиції до кінцевої точки). Якщо ця відстань (перпендикуляр до відрізка) не перевищує радіуса розпізнавання, то відповідний об'єкт вважається обстеженим «по дорозі» без необхідності змінювати траєкторію.

Переваги: простота реалізації, низька обчислювальна складність, швидке знаходження припустимого розв'язку. Недоліки: можливість застрягання в локальному максимумі, відсутність гарантії оптимальності.

Метаевристичний алгоритм орієнтований на побудову маршруту, що забезпечує обстеження максимальної кількості об'єктів за умови обмеження на максимальну довжину польоту БПЛА. Алгоритм є багатоетапним і поєднує: 1) жадібне формування початкового маршруту, 2) локальний пошук у просторі перестановок, 3) геометричне уточнення траєкторії з урахуванням зон обстеження.

1) Побудова початкового маршруту. На першому етапі формується початковий маршрут, який проходить через центри об'єктів. Для цього застосовується жадібний алгоритм. Отриманий маршрут, як правило, не є оптимальним, однак забезпечує охоплення значної кількості об'єктів.

2) Локальний пошук. Далі виконується локальний пошук. Основна ідея полягає у переборі перестановок порядку відвідування об'єктів (за фіксованої їх кількості) з метою мінімізації сумарної довжини маршруту. Це дозволяє покращити початковий маршрут без зменшення кількості обстежених об'єктів.

3) Геометричне уточнення (з урахуванням зон обстеження). Після локального пошуку виконується геометрична оптимізація: кожна проміжна

точка маршруту (окрім стартової та фінішної) зміщується з центра об’єкта до точки на межі зони обстеження цього об’єкта. Зміщення здійснюється таким чином: для точки, що відповідає об’єкту j, розглядається відрізок, який сполучає попередню та наступну точки маршруту; з центра об’єкта опускається перпендикуляр на цей відрізок, і вибирається точка перетину цього перпендикуляра з межею кола радіуса r(зони обстеження). Після коригування всіх проміжних точок обчислюється сумарна довжина уточненого маршруту.

4) Зменшення кількості об’єктів (за потреби). Якщо після оптимізації сумарна довжина маршруту все ще перевищує допустиме значення L, алгоритм зменшує кількість об’єктів у маршруті: виключається один об’єкт — той, який дає найгірший внесок у довжину маршруту (у вашому описі: “найбільш віддалений від лінії між попередньою та наступною точками”). Після цього повторюється весь цикл: локальний пошук і геометричне коригування.

Процес триває доти, доки не буде отримано маршрут, що задовольняє всі обмеження.

Переваги: зниження ризику “застрягання” в локальних оптимумах завдяки поєднанню глобальнішого пошуку та локальних покращень. Недоліки: висока обчислювальна складність порівняно з простими евристиками, відсутність гарантії оптимальності.

Аналіз експериментальних результатів. Метаевристичний алгоритм стабільно демонструє кращі результати за критерієм якості, однак так само стабільно потребує більше обчислювальних ресурсів. На рис. 2 наведено маршрути, побудовані жадібним (ліворуч) і метаевристичним (праворуч) алгоритмами для тестової задачі з $n = 30$.

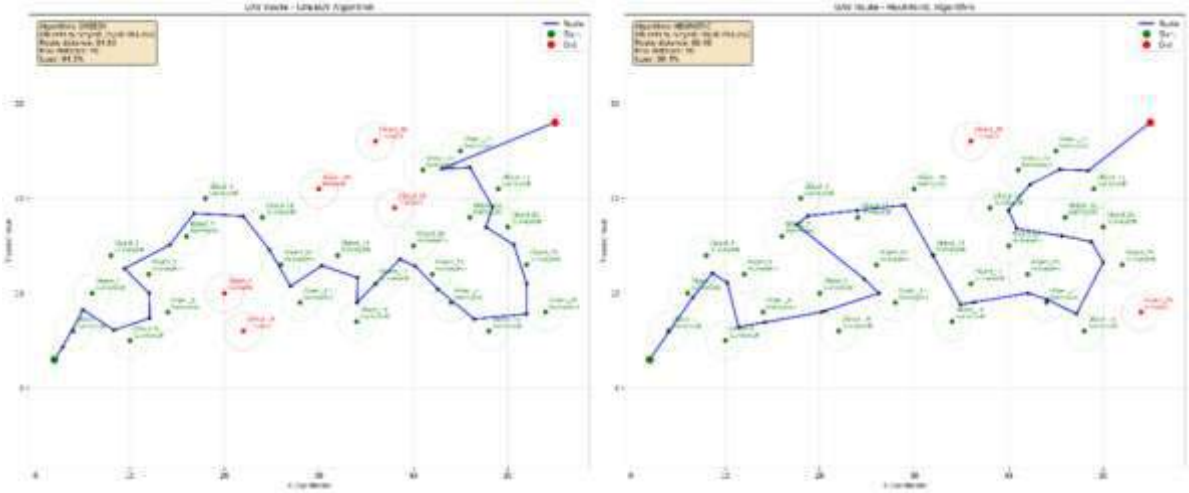

Рисунок 2. Приклад маршрутів БПЛА обстеження однакової множини об’єктів, побудованих жадібним і метаевристичним алгоритмами

Необстежені об’єкти позначено червоним кольором. Як видно з рис. 2, за «жадібним» маршрутом не вдалося обстежити п’ять об’єктів, тоді як за «метаевристичним» — лише два.

Досліджувався вплив параметрів задачі на порівняльні характеристики алгоритмів (покриття об’єктів обстеженням, час виконання), зокрема: кількість

об’єктів, радіус обстеження, щільність розміщення об’єктів (міра їх згрупованості в кластери). З проведених розрахунків випливає, що у випадках, коли обмеження на довжину маршруту є активним, метаевристичний алгоритм обстежує на 25–45% більше об’єктів. Водночас жадібний алгоритм у середньому виконується швидше за метаевристичний приблизно на 1,5 порядки. Ця різниця зростає до трьох порядків у разі збільшення кількості об’єктів до 40–50, що робить жадібний алгоритм більш придатним для задач із обмеженими обчислювальними ресурсами або вимогами швидкої обробки.

Список використаних джерел

1. A. Barnawi, K. Kumar, N. Kumar, N. Thakur, B. Alzahrani, and A. Almansour. 2023. Unmanned aerial vehicle (UAV) path planning for area segmentation in intelligent landmine detection systems. Sensors, vol. 23, no. 16, p. 7264.
2. S. Ghambari, M. Golabi, L. Jourdan, J. Lepagnot, and L. Idoumghar. 2024. UAV path planning techniques: A survey. RAIRO - Operations Research, vol. 58, no. 4, pp. 2951–2989.

VILLAGER FRAMEWORK – A NEW CYBERTHREAT BASED ON ARTIFICIAL INTELLIGENCE

Kamianskyi Yaroslav
Second year cadet
ORCID ID : 0009-0000-5035-6441
Lytvynenko Roman
Second year cadet
ORCID ID : 0009-0007-8327-3803
Riabets Viktor
Second year cadet
ORCID ID : 0009-0007-4746-3554
Scientific supervisor:
Kalyakin Serhiy Volodymyrovych
Senior Lecturer
ORCID ID : 0000-0003-3946-0189
Department of Counteracting Cybercrime
Institute No. 4
Kharkiv National University of Internal Affairs

Villager is an artificial intelligence-based penetration testing system developed by a Chinese group. The system, introduced as Red Team solution designed to fully automate security verification tasks using the MCP protocol, where the combination of the Kali operating system Linux and DeepSeek learning models create a set of advanced automation features.[1] With its easy-to-use functionality, Villager was able to follow in the footsteps of tools like Cobalt Strike, which were originally developed