

Improving UAV Group Reference Routes Using a Stack of Iterative Local Optimization Methods with Preliminary Clustering of Route Points*

Viktor Shevchenko^{1,*†}, Yevhen Derevianko^{1,†}, Valerii Chystiakov^{1,†}, Leonid Artiushyn^{2,†}, and Volodymyr Shevchenko^{3,†}

¹ Institute of Software Systems of the National Academy of Sciences of Ukraine, 40 Academician Glushkova Avenue, Building 5, Kyiv, Ukraine, 03187

² State Research Institute of Aviation, Kazarmenna St., 6V, Kyiv, Ukraine, 01135

³ University of Sheffield, Sir Frederick Mappin Building, The University of Sheffield, Mappin Street, Sheffield, S1 4DT, United Kingdom

Abstract

The development of operational computational procedures, models and methods for finding the best routes for single UAVs and UAVs in a group in emergency situations is an urgent task. The quality criterion is the cost of passing individual sections of the route, which depends on the distance between the route points, on the energy, fuel and depreciation costs of the UAV, which depend on the distance, on the probability of dangerous situations: for the UAV, or due to the activity of the UAV, or regarding the failure to complete tasks, etc. To meet the conditions for the accuracy and speed of forming UAV group routes, a combination of methods for creating reference routes and methods for their further improvement was used. Methods were proposed that work equally effectively for both single UAVs and for a UAV group. In emergency situations, a UAV group is used to perform a large number of tasks. Optimization of the route of each UAV is performed taking into account the routes of other UAVs in the group. Taking into account the limitations on computing resources, the solution is suboptimal. The proposed approach was tested in the MatLab software environment. The method of refining the reference solution using a stack of methods for 15 UAVs provided a solution in 163.75 seconds with an improvement in the route cost by 10.73%. When the number of UAVs increased by 15 times, the computational costs increased by 8.27 times, and the refinement of the reference solution decreased by 1.37 times. It was determined that the computational costs depend almost linearly on the number of UAVs. In the experiment on creating and improving reference routes for 9 UAVs (50 route points), the preliminary distribution of route points between UAVs using the farthest neighbor clustering method required 11% more time, but provided a 9% better result in terms of route cost. The preliminary distribution of route points between UAVs using the nearest neighbor clustering method, which showed the best results, did not require more time, but improved the result by 21%.

Keywords

emergency situation, UAV group, flight route, route points, optimization, reference solution, model, computational methods

1. Introduction

Unmanned aerial vehicles (UAVs) are widely used in conditions that are dangerous for humans. Compared to humans, UAVs have better characteristics in many aspects: speed, movement in a difficult environment, the ability to move cargo, continuous operation time without fatigue, protection from dangerous factors (temperature, chemicals, radiation, lack of oxygen for breathing, mechanical damage, etc.), resistance to factors that interfere with or distract from the performance

2nd Workshop "Intelligent information technologies" UkrProg-IIT'2026 co-located with 16th International Scientific and Practical Programming Conference UkrPROG'2026, May 5-6, 2026, Kyiv, Ukraine

* Corresponding author.

† These authors contributed equally.

✉ gii2014@ukr.net (Viktor Shevchenko); evg.derevjanko@gmail.com (Y. Derevianko); val.rus.ua@gmail.com (Valerii Chystiakov); artleonid2017@gmail.com (Leonid Artiushyn); vshevchenko1@sheffield.ac.uk (Volodymyr Shevchenko)

ORCID 0000-0002-9457-7454 (Viktor Shevchenko); 0000-0003-2949-8896 (Y. Derevianko); 0009-0000-6019-8352 (Valerii Chystiakov); 0000-0002-7488-7244 (Leonid Artiushyn); 0000-0002-2152-6816 (Volodymyr Shevchenko)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

of tasks (threat of loss or damage to the UAV), etc. UAVs are widely used to move cargo, monitor objects, relay signals, create periodic telecommunication networks, etc. The high mobility and diverse functionality of UAVs allows them to be used to perform various tasks related to rescuing people and eliminating danger in emergency situations and in military affairs [1-4]. The task of effective use of UAVs in emergency situations is relevant.

When eliminating emergencies, new tasks and, accordingly, new route points appear suddenly at unpredictable moments in time. This requires replanning UAV flight tasks in real time. In an emergency, the task must be solved on available computing resources that can actually be delivered to the emergency zone. The main component of planning is the route, which is optimized according to many criteria. Computational methods are needed to find routes that are optimal in terms of cost, which take into account the distance, fuel consumption, charge and equipment resource, which depend on the distance, the probability of damage or loss of the UAV, and other hazards. The task of operational optimization of the UAV route using conventional business-class computing resources is relevant. In emergency situations, the number of tasks and, accordingly, route points can be very large. To work out a large number of route points, several UAVs must be used. The previous task is transformed into the task of quickly finding optimal routes for UAVs in a group for a given set of route points and tasks using conventional computing tools.

Thus, the development of quick computing procedures, models and methods for finding the best routes for single UAVs and UAVs in a group in emergency situations is a relevant task.

2. Analysis of existing research

The use of a group of UAVs is considered in [3, 5, 6]. Unfortunately, the main attention was paid to group flight, and not to the simultaneous execution of UAV tasks at different points of the route and not to the traveling salesman problem. Sales Man Problem is the problem of finding the best route, when it is necessary to visit each of the possible points of the route once and return to the starting point [7, 8]. The quality criterion is the minimum cost of the route, which is the sum of the costs of all sections of the route.

The easiest way to solve the traveling salesman problem is using greedy algorithms, when at each step of the search the best solution is chosen, without paying attention to the overall picture of possible solutions as a whole [9-14]. Greedy algorithms can operate at the global level - to build the route as a whole and at the local level - to improve individual sections of the route. The main global greedy algorithms are:

Nearest Point Method at each iteration step adds the nearest point to the route.

Nearest Insertion Method builds a guaranteed smallest traveling salesman cycle for a small number of points. Then adds the closest point to the cycle, which is inserted into the cycle in such a place as to ensure minimal growth in the cycle length.

Cheapest Insertion Method at each iteration step checks all possible options for inserting each new point into each possible place of the already assembled cycle.

Farthest Insertion Method, similarly to Nearest Insertion Method, builds a guaranteed smallest traveling salesman cycle for a small number of points. Then adds the farthest point from the cycle, which is inserted into the cycle in such a place as to ensure minimal growth in the cycle length.

The advantage of greedy algorithms is accuracy and practically no restrictions on the number of route points. The disadvantage is low accuracy. On routes from 50 to 200 points, the error can reach 25-45%. In addition, greedy algorithms can form locally suboptimal sections.

Brute Force is in a sense the opposite of greedy algorithms, as it is aimed at finding the exact optimal solution by exhaustively searching through all possible options [15-19]. The advantage of the method is an exact optimal solution. The disadvantage is a large expenditure of computational resources. The method is effective when the number of route points is no more than 10-15.

The Bellman discrete dynamic programming method [20] is sometimes defined as a method of ordered exhaustive search of all relevant options, which allows a certain proportion of route options to be discarded as unpromising [21]. This significantly reduces computational costs. The method is

implemented in the form of the Held-Karp algorithm [22, 23]. The advantage of the method is an optimal route. The disadvantage is a large expenditure of memory and a limitation of the number of possible route points at the level of 20-25. The Bellman method can be used not only to optimize the route of a traveling salesman passing through specific route points, but also to cover certain territories with a given level of overlap of sensor surveillance zones [24, 25]. The method easily takes into account additional quality criteria, such as minimum energy consumption, minimum cost, and maximum coverage of the territory by onboard sensors.

However, some studies [26, 27] conclude that the high computational complexity of elegant mathematical and algorithmic methods, such as the Bellman method, may hinder their use, regardless of good convergence. To overcome the contradiction between accuracy and computational costs, it is proposed to first search for a reference (suboptimal) solution under simplified conditions instead of the optimal solution and then improve it in an iterative procedure. At the same time, at any step it is possible to exit the iterative procedure to ensure the desired balance between accuracy and computational costs. Exit occurs when one or more conditions are met: achieving a given accuracy, stopping route improvement, reaching a limit number of iterations, etc. The use of methods for improving reference solutions in cloud systems is discussed in [28]. Improving reference solutions for optimizing the control of metal balls on a physical playing field in the well-known Pin Ball game is discussed in [29]. Greedy algorithms, which have high performance, are often used to create a reference route. Then, reference route improvement (RRI) methods are used [26, 30, 31].

When forming UAV routes, a change in UAV task settings in emergency situations is highly probable and rapid, which can occur even during the UAV flight to the task area or between the areas of performing different tasks. Changes in task settings are possible due to the appearance of new or disappearance of old route points after clarification of information on the localization and condition of affected people and objects, or due to the failure of individual UAVs, etc. RRI methods are appropriate because their probable speed is higher than the probable speed of changing UAV task settings in emergency situations. To meet the conditions for the accuracy and speed of forming UAV group routes, a combination of methods for creating reference routes and methods for their further improvement is appropriate. Preference is given to methods that work equally effectively for both single UAVs and UAV groups.

The aim of the article is to improve the quality of UAV group route creation by quickly constructing a reference solution and its further improvement through the cyclic application of a stack of special iterative methods.

3. Route optimality criterion

The quality criterion is the minimum cost of the route. The cost of passing individual sections of the route depends on the following indicators: D_e – Euclidean distance between a pair of route points, E – battery and/or fuel energy consumption, F – UAV depreciation, q_{forUAV} – probability of damage or danger to the UAV, $q_{fromUAV}$ – probability of creating or aggravating dangerous situations due to the UAV, q_{task} – probability of failure to complete tasks, etc. The main component in the cost of the section between points i, j is the Euclidean distance

$$Cost(i, j) = D_e(i, j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \quad (1)$$

where x, y are the coordinates of the points. The second and third indicators of the cost of route sections E, F with coefficients k_E, k_F linearly depend on the length of the route

$$Cost_{D_e, E, F}(i, j) = (k_E + k_F) D_e(i, j) \quad (2)$$

The indicators $q_{forUAV}, q_{fromUAV}, q_{task}$ with a scale factor of 100 are combined into

$$k_{Hazard}(i, j) = 100 (1 - (1 - q_{forUAV})(1 - q_{fromUAV})(1 - q_{task})) \quad (3)$$

To take into account the hazard, a distance multiplier dependence based on the logistic dependence has been proposed [21] (Figure 1).

$$SL(k_{Hazard}(i, j)) = d + \frac{a - d}{1 + e^{-\frac{2}{T}(k_{Hazard}(i, j) - s)}}, \quad (4)$$

where a, d are asymptotes, T is the constant of the argument of the logistic dependence, s is the point of symmetry.

Taking into account all indicators, the cost of the route section is equal to

$$Cost_{\Sigma}(i, j) = SL(k_{Hazard}(i, j)) (k_E + k_F) D_e(i, j). \quad (5)$$

The cost of the route is equal to the sum of the costs of all route sections.

$$CostRoute = \sum_{\substack{RouteSet, \\ i=2, N}} Cost_{\Sigma}(i, i - 1). \quad (6)$$

where $RouteSet$ is the set of route point numbers, N is the number of route points.

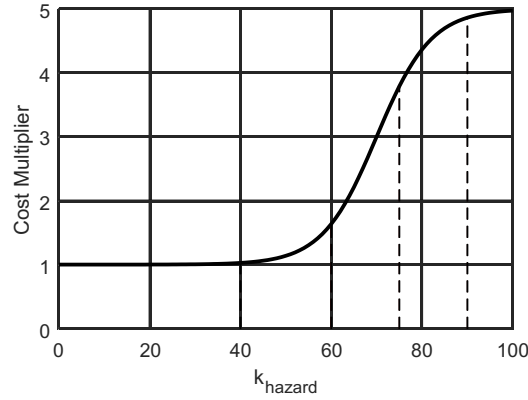


Figure 1: Dependence of the cost multiplier.

4. Finding a suboptimal route

4.1. Creating a reference route

The reference route is the first approximation in solving the problem. Iterative procedures are based on this route and at each iteration bring new route options closer to the optimal one. The reference route is found by the greedy Nearest Point Method algorithm.

Algorithm for forming the reference route.

0. The starting point (x_0, y_0) is selected

Iteration 1.

1.1. We write the point number (x_0, y_0) into the ordered set of route points $RouteSet=[0]$.

1.2. From the previously calculated array, we take the costs of the sections between this point and all other route points $Cost_{\Sigma}(0, j)$, $j = \overline{1, N}$.

1.3. We find the point to which the distance is the smallest

$$Cost_{\Sigma min}(1) = \min_{j=1}^N Cost_{\Sigma}(0, j). \quad (7)$$

1.4. We write the number that is stored for this point in the original (unordered) array of all route points $LocationSet$, into the ordered set of route points $RouteSet$.

1.5. We delete the selected point from the *LocationSet* array.

Then we take this selected point as the initial one and repeat the procedure. The corresponding dependencies for iteration number i have the form:

Iteration number i :

2.1. From the incomplete route *RouteSet* we take the last point (x_{i-1}, y_{i-1}) .

2.2. From the previously calculated array we take the costs of the sections between this point and all other route points $Cost_{\Sigma}(i-1, j)$, $j = \overline{1, N}$.

2.3. We find the point to which the distance is the smallest

$$Cost_{\Sigma min}(i) = \min_{j=LocationSet} Cost_{\Sigma}(i-1, j). \quad (8)$$

2.4. Write the number that is stored for this point in the *LocationSet* array to the ordered set of route points *RouteSet*.

2.5. Delete the selected point from the *LocationSet* array.

2.6. If the iteration number $i < N - 1$, go to step 2.1.

2.7. If the iteration number $i = N$, then write the last two route points [*LocationSet*(1) 0] to *RouteSet*. Note that at this point only one element remains in the *LocationSet* array.

3. The formation of the reference route is completed.

In most cases, the quality of the reference route is not satisfactory. Therefore, it is necessary to improve the reference route using a set of local optimization based on local variation methods, which are placed in the stack. Each method is repeated several times. The entire stack of methods is also repeated several times until the condition for terminating the iteration procedure is met.

4.2. Improving the reference route

The success of improving the reference route depends on the set and order of methods in the stack [32]. In this study, the following methods were found to be the best in the stack.

1. 1PM - 1-Point Moving Method - a method of moving 1 point to an arbitrary new position.

2. 2PE - 2-Point Exchange Method - a method of exchanging 2 points with their places on the route.

3. DC - Delete Crossing Method - a method of eliminating intersections of route sections.

Given that everything starts with creating a reference route using the Nearest Point Method (NPM), a typical RRI scenario looked like this.

$$NPM + (2PE * r_M + 1PM * r_M + DC * r_M) * R_{Stack}, \quad (9)$$

where $4 \leq r_M \leq 9$ – maximum number of repetitions of a single method,

$7 \leq R_{Stack} \leq 10$ – maximum number of repetitions of the method stack.

The improvement of the reference route was performed using the following algorithm.

1. Load the stack of methods of local route variations. Assign the initial value to the method number in the stack.

2. Start the iteration cycle of the method stack.

2.1. Select the RRI method by the current number from the stack.

2.2. Start the iteration cycle of the current RRI method.

2.3. Execute the current RRI method (Iterative procedure for enumerating all possible local route variations within the current method)

2.4. Checking the condition for stopping the cycle of repetitions of the current RRI method (no cost improvements during 2 iterations). If the condition is met, then cyclic incrementing the current RRI method number in the stack and going to step 2.1.

2.5. Checking whether the maximum allowable number of repetitions of the current RRI method r_M has been reached. If the condition is not met, then returning to step 2.3.

2.6. Exiting the RRI method cycle.

3. Checking the condition for stopping the cycle of repetitions of the RRI method stack (no cost improvements during $2N_{Method}$ of the last iterations of the methods, where N_{Method} is the number of methods in the stack). If the condition is not met, then cyclic incrementing the RRI method number in the stack and going back to step 2.1.

4. Checking whether the maximum allowable number of repetitions of the method cycle R_{Stack} has been reached. If the condition is not met, then the cyclic increment of the RRI method number in the stack and return to step 2.1.

5. Exit from the method stack cycle.

6. Refining the reference route is complete.

The condition for stopping the RRI iterative procedures for iterative cycles of local variation methods has the form

$$CostRoute_i(end) = CostRoute_i(end - 1), \quad (10)$$

where $i = \overline{1, N_{Method}}$ – method number in the stack, end – number of the last element (last iteration) in the array of route values by refinement iterations.

The stopping condition of the iterative procedures of the RRI for iterative cycles of the method stack with N_{Method} elements has the form

$$\begin{aligned} CostRoute_i(end) &= CostRoute_i(end - 1) = \\ &= CostRoute_{i+1}(end) = CostRoute_{i+1}(end - 1), \quad i = \overline{1, N_{Method} - 1} \end{aligned} \quad (11)$$

In the first case, there is a cyclic transition to the next stack method, in the second, the search is completed.

5. Testing methods and discussing results for finding optimal routes for a group of UAVs

To perform all tasks in emergency situations, one UAV may not be enough. Usually a group of UAVs is needed. The optimization of the route of each specific UAV in the group should be performed taking into account the routes of other UAVs.

5.1. Problem 1. 200 points, 1 UAV. Finding the optimal route as a preparation for optimizing the routes of a group of UAVs

Consider a task in which it is necessary to perform a task (delivery, return of goods, monitoring) in 200 points. Exact methods do not work with such a number of points on conventional computers. The method of refining the reference solution using a stack of methods for one UAV using an Intel Core i7-10 processor, 16 GB RAM, provided a solution in 19.80 seconds (Figure 2). When using the stack of methods [1 2 3 2 1 3 1 3 2], an improvement in the route cost from 22629 to 19307 (14.68%) was achieved in 38 productive iterations (which led to an improvement in the result). One application of one method was considered as 1 iteration, which could be applied several times within the procedure. After 6 iterations without improving the result, the iterative procedure was stopped.

5.2. Problem 2. 200 points, 15 UAVs.

The computational complexity of the problem of finding optimal routes for a group of UAVs is significantly higher compared to the problem of optimizing the route of a single UAV. For a group of UAVs, the approach with refinement of the reference trajectory is even more appropriate than for a single UAV. Otherwise, the maximum possible number of route points will be even smaller. Therefore, to find the optimal route of a group of UAVs, which consists of the routes of individual UAVs included in the group, the same approach was used as for optimizing the route of a single UAV.

It is clear that one UAV is not able to process such a large number of points even if the tasks at the route points (delivery, return, monitoring) are very simple and do not take much time (the fastest

are monitoring tasks). Therefore, a group of 15 UAVs was involved in performing the tasks. For a similar set of 200 points, the task of optimizing routes for 15 UAVs was performed using the same methods as for one UAV. The main difference was that the entire set of route points at the beginning of the procedure was approximately equally divided between all UAVs with a difference of no more than 1 point. During iterative refinement, individual route points could move from one UAV to another. The routes of all UAVs began and ended at the same start point. This led to a significant increase in the total cost of the route due to the fact that each UAV had to fly two additional sections from the start point and back (a total of $(15-1)*2=28$ sections more).

The method of refining the reference solution using a method stack for 15 UAVs (Intel Core i7-10 processor, 16 GB RAM), provided a solution in 163.75 seconds (Figure 3). With the same sequence of methods in the stack [1 2 3 2 1 3 1 3 2], an improvement in the route cost from 59408 to 53034 (10.73%) was achieved in 72 productive iterations.

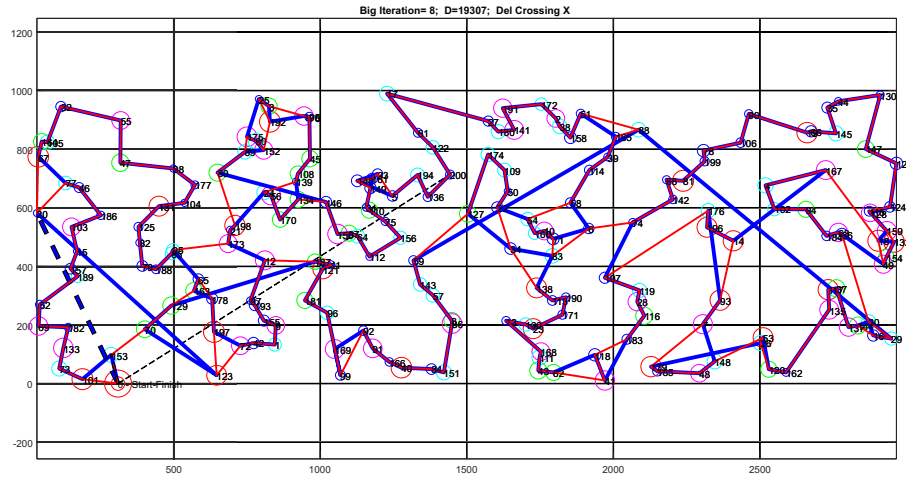


Figure 2: Result of improving the reference solution using a stack of iterative methods for 1 UAV and 200 waypoints.

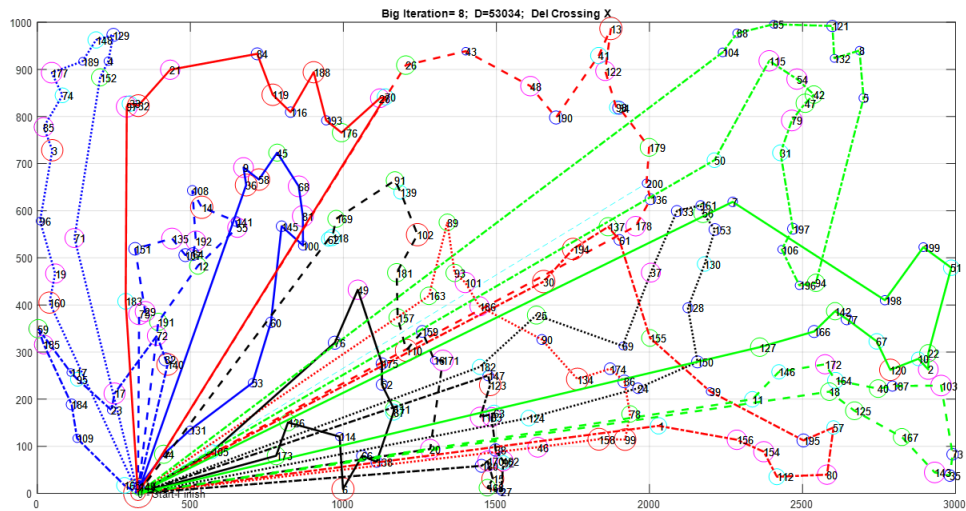


Figure 3: Result of improving the reference solution using a stack of iterative methods for 15 UAVs and 200 waypoints.

As we can see, with an increase in the number of UAVs by 15 times, the computational costs increased by almost an order of magnitude (by 8.27 times), and the refinement of the reference solution decreased by 1.37 times. Taking into account the increase in the complexity of the problem, the increase in computational costs, it can be assumed that the computational costs depend almost linearly on the number of UAVs.

5.3. Problem 3. Finding optimal routes for a group of UAVs with preliminary clustering of route points.

To improve the quality of the route, preliminary clustering of route points was used. The results were compared with experiments without preliminary clustering. The conditions for performing experiments to study the feasibility of preliminary clustering of route points were as follows: 50 pseudo-random route points, 9 UAVs, stack of methods [3 2 2 2 2 3] (the best for such a problem statement).

Experiment 3.1. Uniform distribution of route points between UAVs without clustering. Calculation time 4.052509 sec. Improvement of the route cost from 58347 to 35760 (Figure 4).

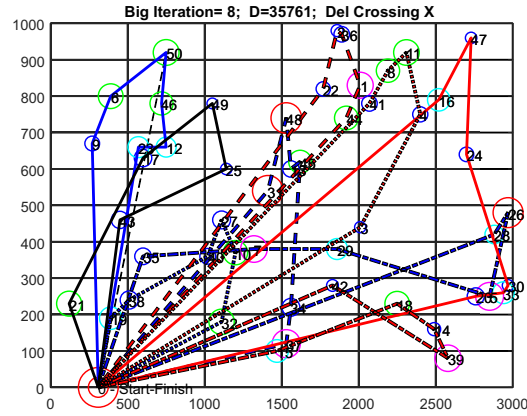


Figure 4: Improving the reference solution when evenly distributing 50 route points between 9 UAVs.

Experiment 3.2. Preliminary distribution of route points between UAVs using the farthest neighbor clustering method.

Calculation time 4.517141 sec. Improvement of route cost from 54848 to 32369 (Figure 5). The method takes 11% more time, but provides a 9% better result in terms of route cost. The qualitative picture of the routes is also more understandable and uniform.

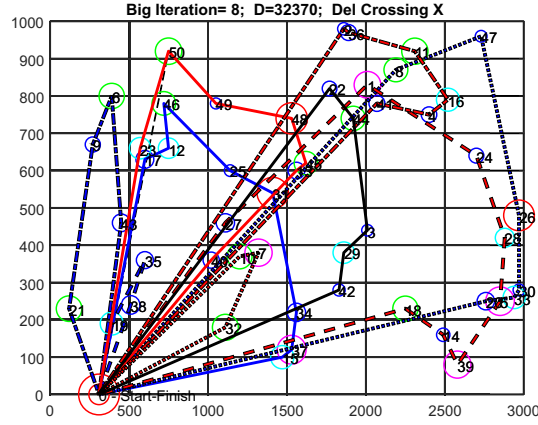


Figure 5: Improving the reference solution when distributing 50 route points between 9 UAVs using the farthest neighbor clustering method.

Experiment 3.3. Preliminary distribution of route points between UAVs using the nearest neighbor clustering method.

Calculation time 4.006244 sec. (almost the same time as in the method of uniform distribution of route points between UAVs without preliminary clustering). Improvement of route cost from 50007 to 29516 (Figure 6). The method provides a 21% better result in terms of route cost. The qualitative picture of the routes is more confusing than in the method of the farthest neighbor, but more understandable than in the method of uniform distribution of route points between UAVs.

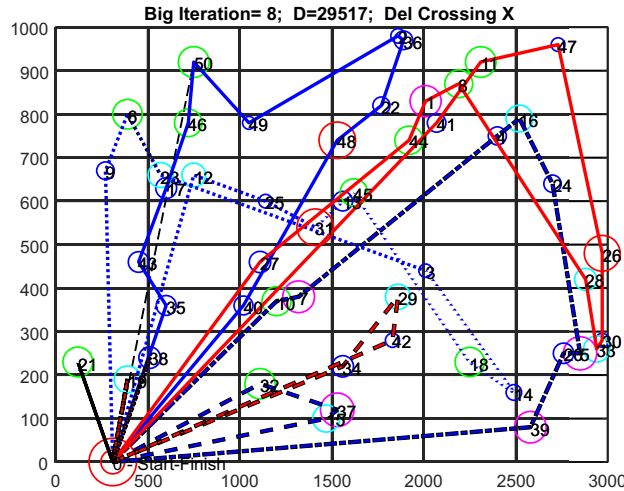


Figure 6: Improving the reference solution when distributing 50 route points between 9 UAVs using nearest neighbor clustering.

The choice of a particular method of distributing route points between UAVs takes into account all the features of the problem statement. The most balanced in terms of the number of tasks is the method of evenly distributing route points between UAVs. The most balanced in terms of the cost of routes of different UAVs is the preliminary clustering of route points using the farthest neighbor method. The preliminary clustering of route points using the nearest neighbor method is optimal in terms of computing resources and the cost of the total route. The disadvantage is the uneven distribution of tasks between different UAVs. This is a payment for reducing the total route length.

6. Conclusions

To meet the conditions for the accuracy and speed of UAV group route formation, a combination of methods for creating reference routes and methods for their further improvement was used. At the same time, methods were proposed that work equally effectively for both single UAVs and for a group of UAVs.

In emergency situations, single UAVs have limited capabilities compared to the scale of the tasks. A group of UAVs is needed to complete all tasks. Optimization of the route of each UAV in the group must be performed taking into account the routes of other UAVs. Taking into account the limitations on computing resources, the solution will be suboptimal, i.e. close to optimal.

The proposed approach was tested in the MatLab software environment.

The method for refining the reference solution using a stack of methods for 15 UAVs provided a solution in 163.75 seconds. An improvement in the route cost by 10.73% was achieved.

Increasing the number of UAVs by 15 times increased the computational costs by 8.27 times, and reduced the value of the refinement of the reference solution by 1.37 times. Taking into account the increase in the complexity of the problem, the increase in computational costs, it can be assumed that the value of computational costs depends almost linearly on the number of UAVs.

In an experiment on creating and improving reference routes for 9 UAVs (50 route points), the pre-allocation of route points between UAVs using the farthest neighbor clustering method took 11% more time and provided a 9% better result in terms of route cost. The pre-allocation of route points between UAVs using the nearest neighbor clustering method, which showed the best results, did not take more time, but provided a 21% improvement in the result.

The directions of further research are to study the capabilities of artificial intelligence and machine learning to find methods that best perform the task of pre-clustering route points between UAVs in a group [33, 34].

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

- [1] V. P. Gorbulin, S. P. Mosov, Unmanned military aircraft in the focus of world scientific and technological progress, *Bulletin of the National Academy of Sciences of Ukraine* 11 (2023) 48–56. doi:10.15407/visn2023.11.048.
- [2] V. Ya. Kanchenko, R. V. Karnaushenko, O. O. Klyuchnikov, O. P. Marynoshenko, M. L. Chepur, Unmanned aerial vehicles for radiation reconnaissance and agricultural purposes, Institute for Safety Problems of Nuclear Power Plants, Chornobyl, 2017. URL: <https://isnpp.kiev.ua/bpla-rad-rozvidky-i-silskogospodarskogo-pryznachennya/>. (in Ukrainian)
- [3] S. P. Mosov, Swarming of military drones: realities and prospects, *Collection of Scientific Papers of the Center for Military and Strategic Studies of the National Defense University of Ukraine* 1 (2024) 77–86. doi:10.33099/2304-2745/2024-1-80/77-86. (in Ukrainian)
- [4] S. P. Mosov, M. V. Pogoretsky, S. M. Saliy, O. V. Selyukov, A. L. Feshchenko, Unmanned aviation in military affairs, Interservice, Kyiv, 2019. URL: https://irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?P21DBN=EC&I21DBN=EC_PRINT&S21FMT=fullw_print&C21COM=F&Z21MFN=808515. (in Ukrainian)
- [5] L. M. Artyushin, V. V. Herasymenko, V. V. Koval, Basic conditions for making a decision on the creation of an automated control system for the group use of unmanned aerial vehicles, *Modern Information Technologies in the Field of Security and Defense* 1 (2021) 63–68. doi:10.33099/2311-7249/2021-40-1-63-68. (in Ukrainian)

- [6] L. M. Artyushin, O. A. Kononov, Yu. V. Shmorgun, Method of forming a joint aviation group, *Modern Information Technologies in the Field of Security and Defense* 2 (2019) 49–54. doi:10.33099/2311-7249/2019-35-2-49-54. (in Ukrainian)
- [7] R. M. Karp, Reducibility among combinatorial problems, in: R. E. Miller, J. W. Thatcher, J. D. Bohlinger (Eds.), *Complexity of Computer Computations*, Springer, Boston, MA, 1972, pp. 85–103. doi:10.1007/978-1-4684-2001-2_9.
- [8] S. Lin, B. W. Kernighan, An Effective Heuristic Algorithm for the Traveling-Salesman Problem, *Operations Research* 21 (1973) 498–516. doi:10.1287/opre.21.2.498.
- [9] Salim Bouamama, Christian Blum, An Improved Greedy Heuristic for the Minimum Positive Influence Dominating Set Problem in Social Networks, *Algorithms* 14 (2021) art. 79. doi:10.3390/a14030079.
- [10] Igor Sinitsyn, Yevhen Derevianko, Stanislav Denysyuk, Volodymyr Shevchenko, Optimizing reference routes through waypoint sequence variation in emergency events of natural and technological origin, in: *Proceedings of the 4th International Workshop on Cybersecurity Providing in Information and Telecommunication Systems (CPITS 2024)*, CEUR-WS.org, 2024, pp. 505–512. URL: <https://ceur-ws.org/Vol-3654/short21.pdf>.
- [11] Volodymyr Shevchenko, Yevhen Derevianko, Oleksii Korniienko, Viktor Shevchenko, Improving the reference route using a stack of iterative local optimization methods, in: *Proceedings of the Workshop "Software engineering and semantic technologies" (SEST 2025)*, CEUR-WS.org, 2025, paper 13. URL: ceur-ws.org.
- [12] Volodymyr Shevchenko, Yevhen Derevianko, Nikita Krupa, Viktor Shevchenko, Statistical Characteristics of the Reference Route Improvement Procedure Using a Stack of Iterative Methods, in: *Proceedings of the Workshop "Intelligent information technologies" (UkrProg-IIT 2025)*, CEUR-WS.org, 2025, paper 6. URL: <https://ceur-ws.org/Vol-4049/paper6.pdf>.
- [13] D. Szeszler, Sufficient conditions for the optimality of the greedy algorithm in greedoids, *J. Comb. Optim.* 44 (2022) 287–302. doi:10.1007/s10878-021-00833-y.
- [14] Yizhun Wang, Review on greedy algorithm, *Theoretical and Natural Science* 14 (2023) 233–239. doi:10.54254/2753-8818/14/20241041.
- [15] Agung Chandra, Christine Natalia, Performance analysis of optimization methods for solving traveling salesman problem, *Innovative Technologies and Scientific Solutions for Industries* 1 (2021) 69–75. doi:10.30837/ITSSI.2021.15.069.
- [16] Condro Wibawa, Optimalisasi Rute Wisata di Yogyakarta Menggunakan Metode Travelling Salesman Person dan Algoritma Brute Force, *Jurnal Teknik dan Science* 1 (2022) 59–65. doi:10.56127/jts.v1i3.512. (in Indonesian)
- [17] Dika Setyo Nugroho, Ahmad Ilham, Brute force algorithm application for solving traveling salesman problem (tsp) in Semarang city tourist destinations, *Jurnal Komputer dan Teknologi Informasi* 3 (2025) 79–86. doi:10.26714/jkti.v3i1.13957. (in Indonesian)
- [18] Heping Jiang, The 1-1 algorithm for Travelling Salesman Problem, *CoRR abs/2104.13197* (2021). doi:10.48550/arXiv.2104.13197.
- [19] Zakir Hussain Ahmed, Ibrahim Al-Dayel, An Exact Algorithm for the Single-Depot Multiple Travelling Salesman Problem, *IJCSNS International Journal of Computer Science and Network Security* 20 (2020) doi:10.22937/IJCSNS.2020.20.09.9.
- [20] R. Bellman, S. Dreyfus, *Dynamic Programming*, Princeton University Press, 2021. doi:10.2307/j.ctv1nxcw0f.
- [21] Viktor L. Shevchenko, Optimization modeling in strategic planning, *Military and Strategic Research Centre of National Defence University of Ukraine*, Kyiv, 2011, 283 p. URL: https://ceur-ws.org/Vol-3806/S_61_Shevchenko_Syvtskyi.pdf. (in Ukrainian)
- [22] Entesar Alanzi, Mohamed El Bachir Menai, Solving the traveling salesman problem with machine learning: A review of recent advances and challenges, *Artificial Intelligence Review* 58 (2025). doi:10.1007/s10462-025-11267-x.
- [23] A. Parjadis, Q. Cappart, B. Dilkina, A. Ferber, L.-M. Rousseau, Learning Lagrangian Multipliers for the Travelling Salesman Problem, in: P. Shaw (Ed.), *30th International Conference on*

- Principles and Practice of Constraint Programming (CP 2024), Schloss Dagstuhl-Leibniz-Zentrum für Informatik, Dagstuhl, 2024, art. 22. doi:10.4230/LIPIcs.CP.2024.22.
- [24] V. Shevchenko, V. Petrivskyi, O. Bychkov, V. Martsenyuk, M. Bernas, A Method for Maximum Coverage of the Territory by Sensors with Minimization of Cost and Assessment of Survivability, *Applied Sciences* 12 (2022) art. 3059. doi:10.3390/app12063059.
 - [25] V. Shevchenko, V. Petrivskyi, O. Bychkov, O. Pokotylo, Models and Information Technologies of Coverage of the Territory by Sensors with Energy Consumption Optimization, in: *Mathematical Modeling and Simulation of Systems (MODS 2021)*, Lecture Notes in Networks and Systems, Vol. 344, Springer, 2022, pp. 17–30. doi:10.1007/978-3-030-89902-8_2.
 - [26] M. C. Horsch, D. Poole, An anytime algorithm for decision making under uncertainty, in: *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI 1998)*, 1998, pp. 246–255. arXiv:1301.7384.
 - [27] Li Kunmei, Nasser Fard, A Novel Nonparametric Feature Selection Approach Based on Mutual Information Transfer Network, *Entropy* 24 (2022) art. 1255. doi:10.3390/e24091255.
 - [28] Farida Asadova, Annamaria Koczy Varkonyine, Gabor Kertesz, Robert Lovas, A Survey of Usage of Anytime Algorithm in Fault detection in Cloud Systems, in: *Proceedings of the IEEE 21st World Symposium on Applied Machine Intelligence and Informatics (SAMI 2023)*, IEEE, 2023, pp. 69–74. doi:10.1109/SAMI58000.2023.10044521.
 - [29] V. Lymperakis, A. A. Panagopoulos, Buggy Pinball: A Novel Single-point Meta-heuristic for Global Continuous Optimization, in: L. Rutkowski, R. Scherer, M. Korytkowski, W. Pedrycz, R. Tadeusiewicz, J. M. Zurada (Eds.), *Artificial Intelligence and Soft Computing (ICAISC 2022)*, Lecture Notes in Computer Science, Vol. 13589, Springer, Cham, 2022, pp. 245–256. doi:10.1007/978-3-031-23480-4_22.
 - [30] J. Grass, S. Zilberstein, Anytime Algorithm Development Tools, *SIGART Bulletin* 7 (1996) 20–27. doi:10.1145/242587.242592.
 - [31] Shlomo Zilberstein, Using Anytime Algorithms in Intelligent Systems, *AI Magazine* 17 (1996) 73–83. URL: <https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/1232>.
 - [32] E. M. Derevianko, V. L. Shevchenko, Determination of the optimal set of methods for iterative improvement of the reference route, *Problems in Programming* 3 (2025) 19–28. doi:10.15407/pp2025.03.019. (in Ukrainian)
 - [33] S. V. Popereshnyak, Using artificial intelligence technologies to optimize methodology and organization of scientific research, *Problems in Programming* 3 (2025) 91–101. doi:10.15407/pp2025.03.091. (in Ukrainian)
 - [34] S. Popereshnyak, V. Ovcharenko, Y. Novikov, H. Hulak, Detection of intrusions based on text analysis and machine learning methods in the development of information systems, in: *Proceedings of the Workshop on Cybersecurity Providing in Information and Telecommunication Systems II (CPITS-II 2024)*, CEUR-WS.org, 2024, pp. 310–318. URL: <https://ceur-ws.org/Vol-3826/short22.pdf>.