Online USV Re-planning with Embedded Pareto Sets

Kilian Le Gall, Laurent Lemarchand, Catherine Dezan

Online USV Re-planning with Embedded Pareto Sets

Kilian Le Gall
Lab-STICC/UBO
Brest, France
kilian.legall@univ-brest.fr

Laurent Lemarchand
Lab-STICC/UBO
Brest, France
laurent.lemarchand@univ-brest.fr

Catherine Dezan
Lab-STICC/UBO
Brest, France
catherine.dezan@univ-brest.fr

***Abstract*—Autonomous vehicles (AV) are known for their ability to perform challenging or risky tasks that would be difficult for humans. In unpredictable environments, AV missions often require real-time path re-planning, adapting to terrain changes, and balancing conflicting objectives like safety, risk assessment, travel time, distance and energy consumption. We have chosen to focus on Unmanned Surface Vehicles (USV) surveillance missions centered around area coverage and using LiDAR (Laser Imaging Detection And Ranging) or camera imagery.**

To tackle these challenges, we propose an offline/online approach for monitoring missions. It exploits a multi-objective Optimization (MOO) framework. In the offline phase, a set of alternative paths is computed using MOO and archived. The archive is in fact a Pareto front. One of these paths is selected as the initial path for the USV, while the online phase handles dynamic path re-planning in response to encountered obstacles during the mission. The re-planning process efficiently and quickly adapts the drone path by reusing the archived data.

Our experiments involve a standard commercial USV and its LiDAR system. Our results demonstrate the benefits of using pre-computed solutions from the offline phase for dynamic path re-planning during the online phase.

keywords*—*USV Path planning, Embedded decision making, path re-planning, multi-objective optimisation

I. Introduction

Autonomous drones have become highly valuable in performing dangerous, challenging or repetitive tasks [5], [18]. For example, they are usable for defense tasks [19] or environmental missions [24]. USVs offer several key benefits. They are not bound by the constraints that affect human operators, such as temperature, space limitations, or environmental disruptions. Additionally, they excel in environments where human intervention is unsafe. Moreover, they are cost-effective and can operate continuously.

Our project aims to deploy an autonomous drone for port surveillance. Each reconnaissance tour conducted by the drone is referred to as a mission. It involves tracking and adapting a calculated trajectory before the mission. This pre-computed trajectory is defined by a sequence of waypoints. These paths aim to minimize the distance traveled by the drone while also minimizing the area of the port facility that is not under surveillance. The drone is equipped with LiDAR (Light Detection and Ranging) which enables it to detect any anomaly in its environment. This LiDAR also assists the drone in detecting obstacles along its path. In the event that the drone encounters an obstacle, it must navigate around it while continuing its surveillance mission to the best of its ability.

The deployment process is divided into two parts. The first one, offline, pre-computes a set of acceptable trajectories for the drone. One of these solution paths is chosen to launch the mission. Following this is the online phase, which involves the adaptation of the pre-calculated trajectory by the drone embedded resources in its environment. During its tour, the drone collects data about its surroundings to achieve monitoring while avoiding unexpected obstacles.

1) Problem statement: However, in the online phase, the obstacle avoidance solution defined earlier does not take into account the two objectives for which the pre-calculated trajectories are optimized. This avoidance method may lead the drone to pass through areas that it has already monitored or was going to monitor later in its mission. In this case, the path re-planning method increases the distance it travels without reducing the area not covered by its LiDAR. Usual USV re-planning algorithms [2] focus on obstacle avoidance and traveled distance and not on extra criteria such as our covering metric.

2) Contributions of this paper: During the offline part of the project, we pre-compute a set of trajectories before selecting one for the mission. Once the offline phase is completed, these solutions are not exploited further. We have developed a new method that leverages these pre-calculated trajectories to better address the two objectives mentioned earlier. This new method is capable of generating a new set of solutions adapted for collision avoidance, referred to as ”repaired” solutions. Repaired solutions are to be embedded for online efficient re-planning. We detail this new obstacle avoidance method along with a comprehensive evaluation. This evaluation was conducted in comparison with a multi-objective approach and a simple re-planning approach that allows the drone to adjust its trajectory when encountering an obstacle [13]. We also assessed the response time of this new method and identified the parameters that impact the quality of its results and its computation times. Finally, we developed an embedded version of this new method. These evaluations were

based on real-world data from a surface drone used in our laboratory. We also performed our evaluations on multiple port facilities. For obstacle avoidance, our new embedded algorithm is preferred to classical algorithm in 91% of the test cases.

We first detail the background of the various methods and algorithms used during the offline and online phases. Then define the approach for our new obstacle avoidance method is presented next. Finally the various experiments that allow us to evaluate our new method are detailed. These experiments aim to (i) assess the necessity of using a multi-objective approach for our obstacle avoidance problem, (ii) evaluate the quality of the results of our approach compared to a trajectory correction that reuses the algorithms from the offline phase, and (iii) show that the method can be efficiently embedded.

II. Background

A. Mission model

In this paper, we are reusing the model developed in our previous publication [13]. As shown in Fig. 1, the environment in which the online and offline parts operate is defined by a grid, associated to a connected graph. The vertices of this graph cover the area under surveillance. They are potential waypoints, i.e. stages of the mission path. The drone can move from one node to any of its four adjacent cardinal nodes. The drone's detection range allows it to detect potential obstacles within a predefined radius around the node where it is stationed. This range is defined by its LiDAR characteristics, which also enables it to collect data about its surroundings. Each node of the graph that is reachable by the LiDAR along the drone trajectory is considered as covered as shown on the upper left corner of 1. The number of covered nodes is used to measure the covered area.

The trajectory of a mission is defined by a succession of waypoint nodes within the graph. The USV goes from one waypoint to the next following a shortest path between the corresponding nodes.

A mission is defined by a trajectory and the values of the objectives associated with that trajectory. Each mission is a compromise between the two objectives, namely the length of the trajectory (Length) and the uncovered area (Uncov). A mission trajectory always starts on the same waypoint and finishes when the drone is back on his starting point.

In our model, the dynamics of the environment corresponds to modifications of the graph. An arising obstacle is defined by two ending nodes, and degrades the graph by making inaccessible all the points forming the shortest path between these two nodes.

B. Trajectories exploration

1) Multi-objective optimisation approach: The USV performs surveillance by conducting tours, also called a mission. A tour is defined by two objectives: the distance covered (Length) during the tour and the area not covered by the onboard LiDAR of the port installation (Uncov). Those two parameters means that a mission is multi-objectives.

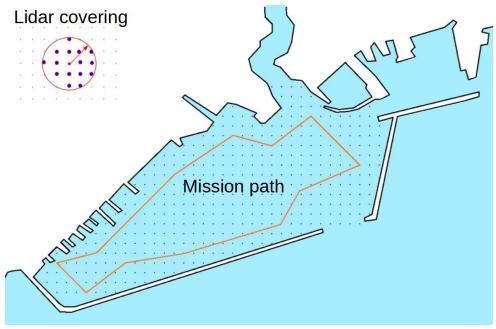

Fig. 1. Path for monitoring mission with covering by a LiDAR

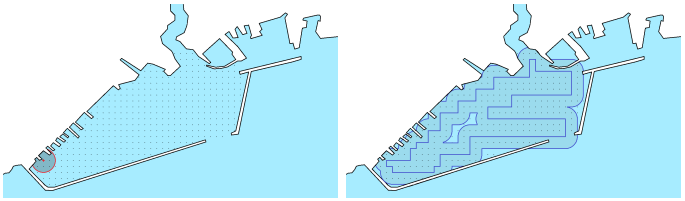

Fig. 2. Example of minimised objective Length (red) and Uncov (blue) in the port facility of Brest.

These two objectives are to be minimized in an optimal case. We aim to minimize the surface area not covered by the drone while minimizing the distance traveled. These objectives are contradictory since reducing the uncovered surface area requires the drone to cover more distance. On the other hand, by reducing the distance traveled, we increase the surface area not covered.

There are two extreme missions, one where the drone does not move to minimize the distance traveled, and one where the drone covers the entire port installation at the expense of the distance traveled. In the two examples presented in Fig. 2, we observe these two extreme cases represented in the port of Brest, red path for the minimisation of Length, and blue for the minimisation of Uncov.

To find the best trajectory, we use a evolutionary algorithm named PAES (Pareto Archived Evolution Strategy)[14]. PAES is a multi-objective optimization algorithm capable of defining a set of solutions close to the Pareto optimal front, as represented by the line in Fig. 3. It maintains an archive of non-dominated solutions. A solution is dominated if another solution is better than the former for all objectives [4]. On contrary a non dominated solution, is a solution without any other solution dominating all its objectives simultaneously. For example, with our two objectives (Length, Uncov), a fictitious trajectory s_1 (14.0,12.0) dominates s_2 (17.4,24.0) because it is shorter than s_2 and covers more surface than s_2.

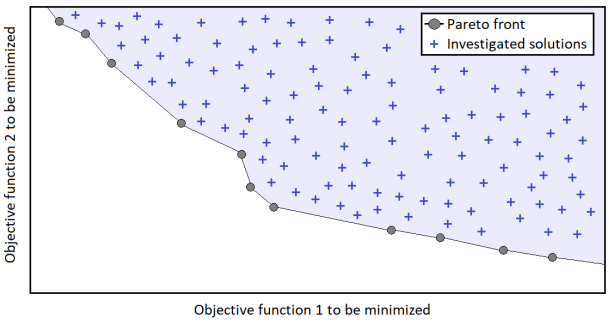

Fig. 3. Example of a Pareto front formed by non dominated solutions.

Every solution over the Pareto front line as seen in Fig. 3 is considered as a bad trade-off and is a dominated solution. The algorithm utilizes its archive of non-dominated (Pareto front) solutions to evaluate the new solutions generated by random mutation. At the end of its search, PAES returns its archive content set.

2) Evaluation and decision: To assess the quality of an archive, a widely used metric is hypervolume [4]. This metric calculates the volume of solution space dominated by a Pareto front, as seen on the upper blue part of Fig. 3. The larger this volume, the more it signifies that the front covers higher-quality solutions. Thus, it also allows for the comparison of two solution sets for the same problem. The hypervolume notably allows observing the evolution of two solution archives. Subtracting the hypervolume of archive A from archive B helps determine if archive A covers more solutions than archive B.

Since PAES returns a set of non-dominated solutions, it is necessary to use decision-making algorithm such as TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) [17] to determine the actual mission to be launched.

TOPSIS is a widely used multi-criteria decision-making method that evaluates alternatives based on their proximity to the ideal solution and their distance from the negative ideal solution. An ideal solution is a solution with the best known value for each objective within a set of solution, even if these values are not associated to a single solution. The anti-ideal is the opposite, with worst known objective values. These two points define the range of values for the objectives associated to the set of solutions.

C. Offline algorithm

The offline part allows for the generation of a trajectory for the mission. It is computed before the mission without any restriction in power and computation time related to an embedded context. To achieve the computation of the trajectory, we have customized the PAES meta-heuristic framework. Our algorithm generates a random trajectory waypoints that it mutates over a predetermined number of generations. We have developed specific mutation operators [13]. At the end of its mutation cycle, PAES provides an archive of non-dominated solutions for the mission trajectory. Each solution present in the archive is therefore considered as an interesting trade-off for the mission.

By applying TOPSIS multi-criteria decision-making algorithm to the archive of solutions, we can identify one of the most suitable trajectories that balance the objectives and constraints for the effective and efficient execution of the mission.

III. Approach

The online part aims to adapt the solution defined during the offline phase for the mission. The drone follows the trajectory and performs supervision tasks. It can detect changes in the environment during its progression. When these changes correspond to obstacles located on its future path, it initiates a repair attempt, i.e a re-planning for the mission. To perform this repair, the drone can rely on a method that we named SPNW (Shortest Path to Next Waypoint), which utilizes Dijkstra shortest path algorithm [9]. Our work presented in this paper aims to develop a new bypass method based on the reuse of the archive produced offline. This new method called RFA (Recovering From Archive) reuses the archive produced by PAES in the offline phase.

A. The current shortest path method

When an obstacle arises, SPNW [13] is used to define a new trajectory that connects the last waypoint reached by the drone to the next waypoint in the trajectory waypoints list. This new trajectory takes into account alterations to the environment caused by obstacles and reported into the graph model. By recalculating the shortest path into this modified graph, the drone can navigate around the obstacle and continue its mission. However, it only considers the length objective function when planning a new path to the next waypoint. Therefore, it is possible for SPNW to make the drone turn back or cross an area that it has already covered or was going to cover in its pre-computed path, as illustrated in Fig. 4.

B. Our proposal : RFA

RFA is our new method, its pseudocode is detailed in the algorithm. 1. It builds upon the archive produced by PAES during the offline phase. It generates a new repaired archive by associating each solution from the archive with the already covered trajectory of the mission. RFA then reuses the multi-criteria selection algorithm to define a new trajectory to complete the mission. By this way, this new trajectory choice takes into account both the length and covering objective functions for re-planning.

In the example shown in Fig. 5, the initial solution in

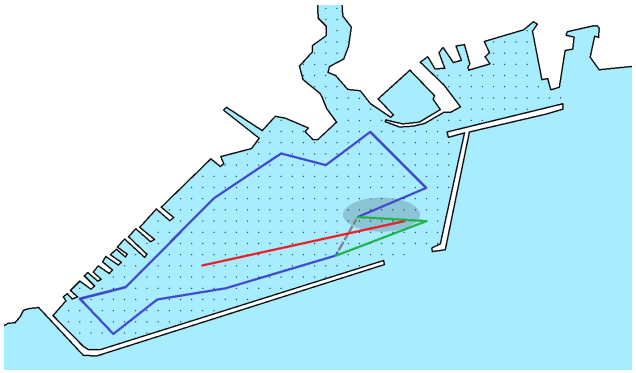

Fig. 4. SPNW producing a obstacle avoidance trajectory

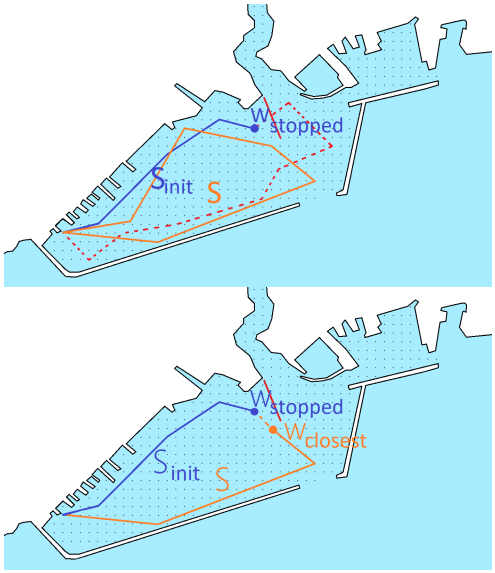

Fig. 5. Repairing initial solution (blue) with RFA by using a solution from the archive (orange)

blue is compromised by the obstacle (red obstacle line and invalidated portion of the path). It is repaired using a solution from the archive, represented by the orange trajectory. The resulting repaired solution, consists of the initial point of the initial trajectory S_{init}, up to the waypoint $w_{stopped}$ before the obstacle. It is then associated with the closest waypoint $w_{closest}$ from the archive trajectory S and continues to follow the remainder of that trajectory back to the initial point.

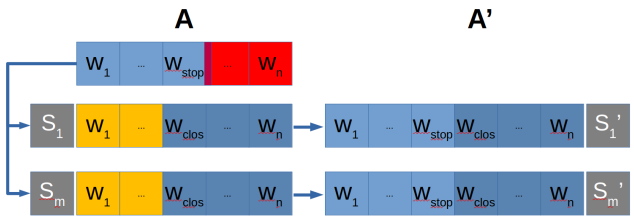

Fig. 6. RFA generation of repaired archive A'

The operation is repeated on each trajectory contained in the archive A to produce the repaired archive A' (Fig. 6). The newly generated solutions are then evaluated using the objective functions Length and Uncov used in the offline phase for the evaluation of solutions produced by PAES. Once A' is filled with the repaired solutions based on A, RFA uses TOPSIS to select a trajectory in A' to achieve the mission.

During the repair process of a solution by RFA, the newly generated trajectory is checked again to ensure it will not also be blocked by the obstacle. To do this, we reuse the shortest path algorithm presented earlier. The solutions in the repaired archive are therefore guaranteed not to go through the obstacle. If some of the solutions present in the initial archive have the same issues as the initial solution, the number of solutions in the initial archive allows for covering a very large sample of different solutions and trajectories. Additionally, the trajectory designed by SPNW is systematically included in the archive since the initially selected trajectory for the mission is used for repair on itself by RFA. This means that since the archive used by RFA for repair also contains the solution used by the drone for its route, the repair attempt by RFA produces a solution based on the initial trajectory, repaired only by SPNW. This will enable us to evaluate SPNW compared to RFA based on our multi-criteria decision algorithm.

Algorithm 1: RFA (Recovering From Archive)

A: Pre-computed archive;
$S_{init} = (w_1, w_2, ..., w_{stopped}, ..., w_n)$: initial solution;
with $w_{stopped}$: waypoint where S_{init} is stopped;
foreach *solution* $S \in A$ **do**
  $w_{closest} = argmin_{w \in S}$ distance$(w, w_{stopped})$;
  $S' = (w_1, ..., w_{stopped}, w_{closest}, ..., w_n)$;
  if S' *encounters an obstacle* **then**
    Correct the trajectory of S' with SPNW;
  end
  Evaluate S';
  $A = A \cup \{S'\} \setminus \{S\}$;
end
Select solution from A with TOPSIS;

Since each solution repair performed by RFA uses the SPNW algorithm, the computation time for our new re-planning strategy is inevitably greater than that of SPNW. This computation time increases linearly with respect to the size of the archive used for the repair. In the case of an archive of size n, e.g. 100, RFA will take at least n times the computation time of SPNW to produce its repaired archive.

The offline PAES archive size is adjusted to the offline computational effort. The larger the archive, the more efficient the exploration process for alternative initial trajectory

computation. However, the exploitation of the embedded archive is constrained by limited computation resources and an execution time budget. Thus, it is possible to decrease the size of the online archive by selecting a subset of elements from the offline archive. There are several methods to achieve this.

PAES adds solutions to its archive while respecting a threshold on its size. It selects solutions according to diversity criteria [14]. We can reuse this mechanism to build the embedded archive. A simpler way consists of sorting the offline archive members according to one of the objective functions and sampling elements regularly with a frequency adjusted to the targeted embedded archive size.

We opted for the second method as we have not yet conducted a study to determine the potential of each solution to repair the trajectory. This way, we ensure to have a sample of solutions that is as representative as possible of our initial archive.

As for the initial path computation, a large embedded archive size allows for better exploration of the space of possible re-planning trajectories, but at the expense of computation time for the repair process. Therefore, it is necessary to find a good trade-off while still meeting the reaction time requirements which depend on the environment in which the USV operates.

IV. Experiments

To assess the usefulness of our RFA method, we first check the results obtained and compare them to a shortest path based collision avoidance algorithm (SPNW) for our objectives. Analysis the results allows to asses the usefulness of MOO for obstacle avoidance during monitoring missions. We then compare the quality and computation time of RFA with a method that reuses PAES for a complete recalculation of the trajectory. Finally, we look at the performances obtained with RFA when embedding it on an ARM processor.

The various experiments presented here follow the same test methodology applied on the same benchmark for path re-planning with SPNW, RFA and PAES.

A. Testing environment and methodology

Our test cases for online re-planning are generated as follows. We first produce an archive offline with PAES for a given supervision mission and select an initial path for the mission using TOPSIS. The USV will follow this path. Next, we generate randomly an obstacle and include it into the routing graph (see Section III). Each test case consists of an archive of solutions produced by PAES in offline, along with a solution selected by TOPSIS that is blocked by an obstacle. The different re-planning algorithms are then used to avoid the obstacle.

To enhance the relevance of the results, we rely on concrete test cases. For this purpose, we utilize data from a commercial Marine Drone 1800 USV [23]. This drone has a maximum speed of 4.2 m/s but typically moves at 1.3 m/s. It has a LiDAR range of 100m. Concerning areas to be monitored, our test cases include three port facilities: Brest, Tropez, and Granville, illustrated in Fig. 7. For each port facility, we generate 200 different scenario, resulting in a total of 600 test cases used as benchmark for our experiments.

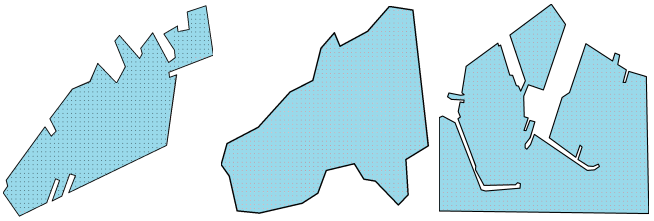

Fig. 7. Map of Brest (left), Tropez (center) and Granville (right) used for our test cases

TABLE I
WAYPOINT DENSITY FOR EACH HARBOR MAP, IN WAYPOINTS/SQUARE KILOMETER

	Brest	Tropez	Granville
waypoint density ($\#waypoints/km^2$)	1731.77	197.17	9290.21

For the grid parameters, we have determined it to be 70 waypoints by 70 waypoints. In the case of Brest, where the width (west-east distance) is 3500m, this places a waypoint every 50m in width. The height of the map (south-north distance) is 2500m, which results in a waypoint every 35m. Each map has the same grid size, but because the maps are not of the same size, the waypoint density is different. Waypoint density of each map is displayed on Table I.

B. RFA vs SPNW

The initial assessment of the new RFA bypass method aims to determine whether a multi-objective approach is necessary for solving our problem. To do this, we compare the solutions generated by RFA with the bypass solutions produced by SPNW for re-planning. SPNW searches for the shortest path to the next waypoint. Thus it doesn't take into account the covering objective of the mission.

We aim to determine if the solutions produced online by RFA have a better Length and Uncov trade-off than the solutions produced by SPNW. To achieve this, we use TOPSIS online to select a solution to complete the mission from the RFA embedded archive. By adding the solution produced by SPNW in the archive before the selection by TOPSIS, we can determine if TOPSIS selects the solution produced by SPNW or by RFA. The origin (RFA or SPNW) of the selected solution indicates which method offers the best trade-off for the test case. The average results for each port (200 test cases per port) in Fig. 8 show the substitution rate (in blue) of RFA as compared to SPNW. It also presents the percentage of RFA dominant solutions over SPNW (in orange). These columns

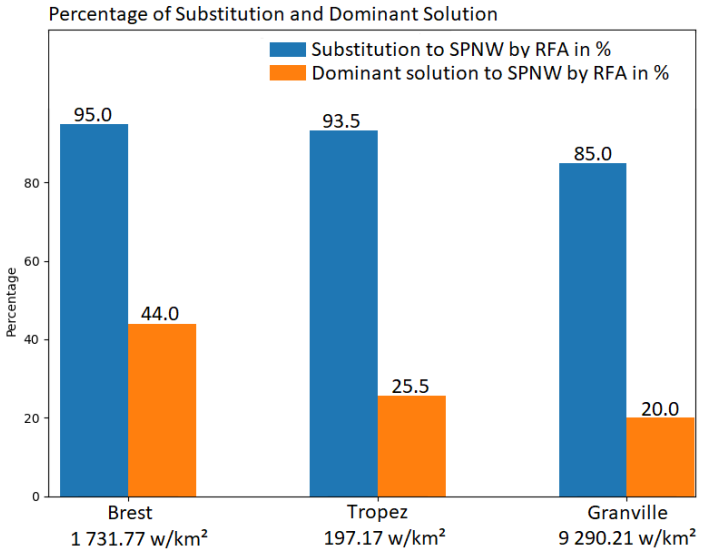

Fig. 8. Substitution percentage of RFA to SPNW on same test case depending on waypoint density (number of waypoint by map size)

of dominant solutions represent cases where the selection algorithm (TOPSIS) selects a trajectory proposed by RFA and this trajectory has lower Length and Uncov values than the solution proposed by SPNW.

We observe that the results are less favorable to RFA for the Granville map test cases (85%) as compared to the Brest and Tropez maps (resp. 95% and 93%). This can be explained by the difference in waypoint density among these different maps. Each map has the same number of waypoints, namely 70*70, which amounts to 4900 waypoints per map. Waypoints density of each map is relayed in Table I.

RFA re-planning is preferred to SPNW for 91% on average of our 600 test cases. This shows the usefulness of the approach for taking into account both Length and Covering objectives during re-planning.

In our previous paper [13], we achieved substitution results of 76%. The variance with our current results can be explained by the improvements made to RFA since our initial work. Additionally, it is mainly attributed to the grid size difference. In our previous work, it was 50*50, whereas here it is 70*70.

Although the computation time of SPNW is much shorter than that of RFA (2.12 ms versus 564 ms), in our real test case using the Marine Drone 1800, RFA produce a correction while the drone has advanced at most by 0.732 m.

C. RFA vs PAES

As shown in previous section, MOO is of interest for re-planning with multiple objectives. However, running a full MOEA algorithm online could be too much resource and time consuming. We check with this second experiment the qualitative and execution time values of PAES (MOEA used in the offline phase) when performing re-planning. Both RFA and PAES produce Pareto sets. In order to compare them qualitatively, we compute the hypervolume value (the higher it is, the best the set is, [4]) with both algorithms for each test case. This way we are able to observe the difference of quality between our two methods (PAES and RFA for online re-planning).

TABLE II
QUALITY LOSS (HYPERVOLUME VALUE REDUCTION) AS COMPARED TO PAES (REFERENCE VALUE FOR HYPERVOLUME, WITH AN ARCHIVE SIZE OF 100) FOR DIFFERENT RFA ARCHIVE SIZES

	PAES	RFA 100	RFA 50	RFA 10
Hypervolume loss	/	14.43%	16.54%	25.69%
Computation time (ms)	53141.56	564.01	373.06	211.86

First row in Table II shows average loss of hypervolume when comparing RFA sets of different sizes to PAES set, considered as the reference set. PAES is run with a fixed archive size of 100. RFA set size is adjusted according to the selection strategy described in Section III. For example, running RFA with a size of archive 100 leads to an average loss of quality compared to PAES (hypervolume value) of 14.43% for our benchmark of 600 test cases as compared to PAES (thus with the same archive size). The quality loss increases to 25.69% with a RFA set size divided by 10.
By analysing the test case results in term of objectives, we observe a slight reduction in the covered area by the archive produced by RFA:100 compared to the archive generated by PAES. This reduction increases while the size of the archive used by RFA decreases.

Additionally, we evaluate the computation times of these two methods to determine the associated computational effort. This corresponds to second row in Table II. Concerning execution times, we note that the computation time of RFA:100 is 94 times faster than that of PAES (reducing from 53141.56 ms (PAES) to 564.01 ms (RFA)). For the RFA:10 method, the computation time is divided by 251, at the expense of the quality of the produced archive. The archive degradation is 14.43% for RFA:100 and 25.69% for RFA:10.

In summary, RFA reduces execution times by almost a factor of 100 while decreasing the hypervolume by 15%, compared to a conventional complete repair based on the solution used for the initial routing (PAES). This provides opportunities to meet execution time constraints in larger contexts than PAES while maintaining the optimization of multiple objectives.

Furthermore the set size reduction mechanism can help to adjust the computation budget in order to meet real times constraints. For example, with the Marine drone 1800 moving at 1.3 m/s and detecting an obstacle at 100 m, the USV will move forward the obstacle for 6.89 meters before re-planning

TABLE III
COMPUTATION TIME OF RFA ON ARM PROCESSOR (CORTEX-A15) BY ARCHIVE SIZE

	RFA:100	RFA:90	RFA:80	RFA:70	RFA:60
x86 computation time (ms)	314.2	314.1	306.4	292.1	272.8
ARM computation time (ms)	3802.0	3527.8	3582.2	3381.4	3136.5

	RFA:50	RFA:40	RFA:30	RFA:20	RFA:10
x86 computation time (ms)	251.6	228.5	207.0	187.7	150.9
ARM computation time (ms)	2963.4	2624.4	2396.5	2166.0	1787.2

with PAES while it will progress of 0.73 m with RFA:100 and 0.27 m with RFA:10. If it runs at its maximal speed of 4.2 m/s, the progression with PAES is of 21.8 m, to be compared with the 100 m range of the LiDAR. Even if the execution time values for RFA seem reasonable for the real test case, we have to check them with an embedded platform that integrate a processor ARM for instance. This is the goal of next experiment.

D. Embedding RFA

The next experiment aims to assess the feasibility of implementing our new bypassing method. This has not been conducted on the same test cases as the previous experiments. Here, we have conducted 100 test cases on the map of Brest, both on an ARM processor (Cortex-A15) and on an x86 processor (Intel Core i7-8750).

ARM architecture is very popular in embedded systems, thanks to its cost and energy consumption. Since our goal is to embed RFA, we developed a version for ARM platforms. We then measure and compare the execution times for both ARM and x86 versions for archive sizes from 10 to 100. The benchmark used is a bit different as previously: there are 100 test cases, located at the port facility of Brest. As an ARM platform, we emulated Cortex-A15 via QEMU, using a single core since RFA is not parallelized.

The Table III presents the average computation time results over the 100 test cases for the different archive sizes (100 to 10). Runtime are roughly multiplied by 12 as compared to an Intel Core i7-8750 execution times. The difference in computation time between x86 processor of this experiment and experiment IV-B is attributed to the number of test cases and their differences (This experiment was only conducted at the port facility of Brest). This experiment was conducted on 100 independent test cases unrelated to the previous experiments. Our goal here is to show the variation of computation time of our new method on the two processors.

It leads for our real example of USV to a reaction distance of 4.94 m (resp. 2.31 m) at a realistic drone speed of 1.3 m/s for an archive size of 100.

The implementation used is a direct port of the x86 version to ARM. It is entirely possible to optimize RFA in order to reduce computation time, particularly by running in parallel the archive repair phase. This phase is the most time consuming of RFA (execution time of decision making with TOPSIS is negligible as compared to it). It can be achieved in parallel for each member of the archive, with an expected speedup close to the number of cores used for this phase since data processed for each member are independent.

V. RELATED WORK

Numerous studies have explored Unmanned Surface Vehicle (USV) path planning using various metaheuristic approaches such as Ant Colony Optimization [1], Genetic Algorithms as shown in [8] and [7]. Our focus lies in addressing a multi-objective port surveillance problem. While many Multi-Objective Optimization (MOO) metaheuristics have been proposed (see [4] for an extensive survey), most of these Multiple Optimization Evolutionary Algorithms (MOEA) necessitate substantial computations at each generation and, eventually, extensive memory resources. On the other hand, PAES [14] provides a single solution at each generation, thus facilitating seamless algorithm integration.

In the context of online obstacle avoidance method for a surface drone within a port facility, some previous works has enabled obstacle detection with LiDAR [11], considering only the covering area for the monitoring mission. Nevertheless, there are existing studies on obstacle avoidance through the search for the shortest path [25]. Studies relying on Pareto fronts generated by a genetic algorithm for obstacle avoidance also exist, as demonstrated in [21] [10], but these propose predefined planning scenarios involving pre-detected obstacles. While obstacle avoidance methods using genetic algorithms do exist [22], they do not share our objective of applying the same framework to real-time obstacle avoidance. In this paper, we propose to leverage the results of the offline genetic algorithm to maintain a multi-objective environment during potential obstacle avoidance. This approach avoids a complete re-run of the genetic algorithm as in [16].

There have been prior works on multi-objective frameworks in the field of surface drones and path optimization [1] [21] [8] [7] [16] [10] [25]. However, not all of them employ genetic algorithms to produce solutions. In our approach, we rely on the reuse of an archive of solutions. The paper in [16] addresses the same issue of multiple objectives, performing online trajectory re-planning with additional constraints. However, their focus is on adhering to marine collision avoidance rules rather than optimizing a coverage metric. Furthermore, they do not encounter the computational constraints associated with deploying solutions on an ARM processor, as is the case in our specific scenario. Real-time

avoidance on an ARM processor requires operating under resource-constrained computational power [3]. While our project tackles with multi-objective optimization that have been widely studied and implemented individually, there is no existing work that proposes embedded online solution in the context of the drone mission.

VI. Conclusion and future work

In this paper, we have proposed a new path re-planning technique for obstacle avoidance for an USV during supervision missions. This new method uses a multi-objective approach and relies on pre-calculated solutions during the offline phase. This way, the drone is able to establish a new trajectory efficiently and in real time, while taking into account both Length and Covering path optimisation objectives. We compared this new method to a classical obstacle avoidance approach based on a shortest path algorithm, demonstrating the effectiveness of a multi-objective approach. We also compared it to a repair method that embeds the offline algorithms for re-planning. This allowed us to demonstrate the efficiency of our approach which requires much less computation time. Furthermore, we deployed our new Pareto set based method on an ARM processor to show its embedding potential.

Our next goal is to refine our dynamic obstacle avoidance method. We plan to study which solutions from the initial archive can be the most useful for embedded dynamic re-planning, in relation with the path chosen for launching the mission. We expect this will help reduce RFA computation time by limiting the size of the embedded archive. Furthermore, additional planning criteria such as COLREGS (collision at sea prevention rules [6]) [16] could to be integrated to our method. This requires an extension of our model to include the dynamics of the obstacle [15]. Last, our offline/online approach can be applied with various MOO algorithms that produce Pareto sets such as popular NSGA2 [12], also coupled with other decision making algorithms than TOPSIS, e.g. [20]. The efficiency of the approach should be tested with these components.

References

[1] Lazarowska Agnieszka. Ship's trajectory planning for collision avoidance at sea based on ant colony optimisation. *The Journal of Navigation*, 68(2):291–307, 2015.

[2] Robin T. Bye Ottar L. Osen Anete Vagale, Rachid Oucheikh and Thor I. Fossen. Path planning and collision avoidance for autonomous surface vehicles i: a review. *Journal of Marine Science and Technology*, pages 1–15, 2021.

[3] Emad Ebeid Arne Devos and Poramate Manoonpong. Development of autonomous drones for adaptive obstacle avoidance in real world environments. pages 707–710, 2018.

[4] Gary B. Lamont Carlos A. Coello Coello and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2007.

[5] Yuanqiao Wen Zhe Du Changshi Xiao Liang Huang Chunhui Zhou, Shangding Gu and Man Zhu. The review unmanned surface vehicle path planning: Based on multi-modality constraint. *Ocean Engineering*, 200, 2020.

[6] U.S. Dept. Homeland Security/U.S. Coast Guard. Navigation rules. Technical report, 2010.

[7] Djamel Benazzouz Mohamed Abdessamed Ait-Chikh Hand Ouelmokhtar, Yahia Benmoussa and Laurent Lemarchand. Energy-based usv maritime monitoring using multi-objective evolutionary algorithms. *Ocean Engineering*, 253:111182, 2022.

[8] Jean-Philippe Diguet Djamel Benazzouz Hand Ouelmokhtar, Yahia Benmoussa and Laurent Lemarchand. Near-optimal covering solution for usv coastal monitoring using paes. *Journal of Intelligent and Robotic Systems*, 106(1):24, September 2022.

[9] Al Savvaris Hanlin Niu, Yu Lu and Antonios Tsourdos. An energy-efficient path planning algorithm for unmanned surface vehicles. *Ocean Engineering*, 161:308–321, 2018.

[10] Maro Jeon JaeHak Kim-Soonseok Song Heesu Kim, Sang-Hyun Kim and Kwang-Jun Paik. A study on path optimization method of an unmanned surface vehicle under environmental loads using genetic algorithm. *Ocean Engineering*, 142:616–624, 2017.

[11] Jooho Lee Joohyun Woo and Nakwan Kim. Obstacle avoidance and target search of an autonomous surface vehicle for 2016 maritime robotx challenge. pages 1–5, 2017.

[12] S. Agarwal K. Deb, A. Pratap and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[13] Laurent Lemarchand Kilian Le Gall and Catherine Dezan. Multi-objective optimization for an online re-planning of autonomous vehicles. pages 48–51, 2023.

[14] Joshua D. Knowles and David W. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary computation*, 8(2):149–172, 2000.

[15] Yoshiaki Kuwata, Michael T Wolf, Dimitri Zarzhitsky, Huntsberger, and Terrance L. Safe maritime autonomous navigation with colregs, using velocity obstacles. *IEEE Journal of Oceanic Engineering*, 39(1):110–119, 2013.

[16] Eshan Rajabally Graham Watson Terry Mills Zakirul Bhuiyan Liang Hu, Wasif Naeem and Ivor Salter. Colregs-compliant path planning for autonomous surface vehicles. *IFAC-PapersOnLine*, 50(1):13662–13667, 2017. 20th IFAC World Congress.

[17] Morteza Yazdani Majid Behzadian, S. Khanmohammadi Otaghsara and Joshua Ignatius. A state-of the-art survey of topsis applications. *Expert Syst. Appl.*, 39, 2012.

[18] Tianhe Liu Meng Joo Er, Chuang Ma and Huibin Gong. Intelligent motion control of unmanned surface vehicles: A critical review. *Ocean Engineering*, 280:114562, 2023.

[19] Thomas Pastore and Vladimir Djapic. Improving autonomy and control of autonomous surface vehicles in port protection and mine countermeasure scenarios. *Journal of Field Robotics*, 27(6):903–914, 2010.

[20] R.V. Rao and R.J. Lakshmi. Ranking of pareto-optimal solutions and selecting the best solution in multi- and many-objective optimization problems using r-method. *Soft Computing Letters*, 3:100015, 07 2021.

[21] R. Saravanan S. Ramabalan and C. Balamurugan. Multi-objective dynamic optimal trajectory planning of robot manipulators in the presence of obstacles. *The International Journal of Advanced Manufacturing Technology*, 2009.

[22] R. Smierzchalski and Z. Michalewicz. Modeling of ship trajectory in collision situations by an evolutionary algorithm. *IEEE Transactions on Evolutionary Computation*, 4(3):227–241, 2000.

[23] IM solutions. Marine drone 1800, `https://im-solutions.fr/en/drone-1800/`. webpage visited 10/30/23.

[24] Lizbeth Leonor Paredes Aguilar Mauricio Postigo-Malaga José M. Garcia-Bravo Wonse Jo, Yuta Hoashi and Byung-Cheol Min. A low-cost and small usv platform for water quality monitoring. *HardwareX*, 6:e00076, 2019.

[25] Mengqi Hu Yong Ma and Xinping Yan. Multi-objective path planning for unmanned surface vehicle with currents effects. *ISA Transactions*, 75:137–156, 02 2018.