Encodings for Multi-Objective Free-Form Coverage Path Planning

Lukas Bostelmann-Arp[1 [0000−0003−2951−5051]], Christoph Steup[1 [0000−0001−6936−9760]], and Sanaz Mostaghim[1,2 [0000−0002−9917−5227]]

[1] Otto von Guericke University Magdeburg, Germany
{lukas.bostelmann-arp,christoph.steup,sanaz.mostaghim}@ovgu.de
[2] Fraunhofer Institute for Transportation and Infrastructure Systems IVI, Germany

Abstract. Coverage path planning (CPP) is the problem of determining a path that covers any given area and is mainly applied in the field of robotics. To ensure efficient coverage, objectives like path length, overlaps, and traversal time are considered. However, in certain scenarios, a simplistic total coverage approach may not be optimal, necessitating a trade-off strategy. This paper addresses a novel challenge in CPP: the multi-objective weighted coverage path planning problem, where total coverage is not strictly required but balanced against other objectives and constraints. We present an approach to solve this problem using evolutionary multi-objective algorithms with free-form path representations. The focus lies on comparing different path representations, ranging from polygonal chains to Bézier curves and B-splines, to Non-Uniform Rational B-splines (NURBs). Additionally, we incorporate an overlaid rectangular grid for comparison with a graph-based approach.

Keywords: Multi-Objective Optimization · Coverage Path Planning · Free-Form · Path Representation.

1 Introduction

Nowadays, mobile robots are utilized in various domains: underwater [10,1], on land, for example in fields [15], and in the air as unmanned aerial vehicles (UAVs). In the latter case, they fulfill various tasks, including videography, remote sensing [25], agricultural monitoring [23], and aiding in disaster scenarios such as earthquake [6]. Despite their diverse applications, they share a common challenge: coverage path planning (CPP). These paths must cover the area of interest while considering metrics such as path length, overlaps, curvature, and traversal time, as well as application-specific constraints like no-fly zones (NFZs) and obstacles. However, in some scenarios, a simple total coverage approach is not sufficient. Instead, a trade-off is necessary. This happens in search and rescue scenarios, for example, when searching for a missing person in a large field with limited time and energy resources. In these cases, a value function can be employed to prioritize certain parts of the area. For instance, in UAV search and rescue missions, this function could represent the probability distribution of the

missing person's location. Additionally, there is a need for adaptable, free-form paths that can adjust to any situation. Thus, arises a novel challenge: the multi-objective weighted coverage path planning problem.

Multi-objective evolutionary algorithms (EAs) have been widely applied to solve coverage path planning (CPP) problems. In most of these works, the area is discretized into subregions or a set of waypoints, and the EA is used to optimize the order of traversal. In addition, the majority of papers either focus on a single objective problem or combine multiple objectives into one. For instance, in [13], an agricultural field is clustered into blocks, and the sequence of traversal of these blocks, along with the respective entry and exit points, is optimized. Further, waypoints can be generated by employing domain knowledge [8] or by overlaying a grid and assigning a waypoint to each cell [22]. Only a few papers have used true multi-objective evolutionary algorithms to solve the CPP problem. One such study optimized the traversal of waypoints to minimize energy usage and maximize coverage for an autonomous underwater vehicle inspecting complex structures [10]. In another paper, the energy usage and coverage for an underwater robot are optimized [1]. Moreover, evolutionary algorithms are used for continuous path optimization using a single objective. Use cases include path planning for UAVs [21,12,18] and general mobile robots [11,20,14]. Although these methods employ a continuous space for the waypoints that form a path, the waypoints are typically connected by straight lines. Some studies use smoothing techniques to convert these polylines into free-form paths afterward. Only two works [19,7] utilized actual free-form path representations, specifically B-splines and Bézier curves. There is also a multi-objective approach [24], but it still uses only line segments between waypoints as the path representation. Lastly, two notable papers integrate multi-objective evolutionary algorithms with free-form path planning [5,4]. These studies focus on generating a coverage path for an agricultural field, both with and without prior decomposition. However, only one seed curve is optimized in these approaches, which is then offset to cover the entire field, rather than optimizing the entire path.

The goal of this paper is to employ EAs to address a multi-objective weighted CPP problem using free-form curves. Our major focus is to study the impact of the path representations in the evolutionary multi-objective optimization context. This aspect is fundamental, as the operators and the evaluation functions highly depend on the selected representation. For this paper, we propose abstract path representations at a mathematical level. Such representations offer scalability and can map nearly any path. We explore various options, ranging from simple polygonal chains to more complex Bézier curves, B-splines, and NURBs. Additionally, we implement an overlaid rectangular grid to facilitate comparisons with a discrete graph-based approach. We examine the proposed representations based on their ability to generate a well spread and converged Pareto front. While cardinal B-splines are not as fast as polygonal chains or as divers as the graph-based approach, they provide the best trade-off between runtime and convergence, making them the most promising approach.

2 Problem Statement

An area for the weighted coverage path planning problem consists of three components: a value function, no-fly zones, and an outer border. The value function is discrete and defined over a high-resolution grid, resulting in the matrix $A \in \mathbb{R}^{m \times n}$. Each NFZ is defined by a polygon describing its outer shape. All generated paths within this area must lie within the outer border of the area and cover as much of the value function as possible, while optimizing both path length and smoothness. The selected scenarios are illustrated in Figure 3 and explained in more detail in section 4.

2.1 Path Representations

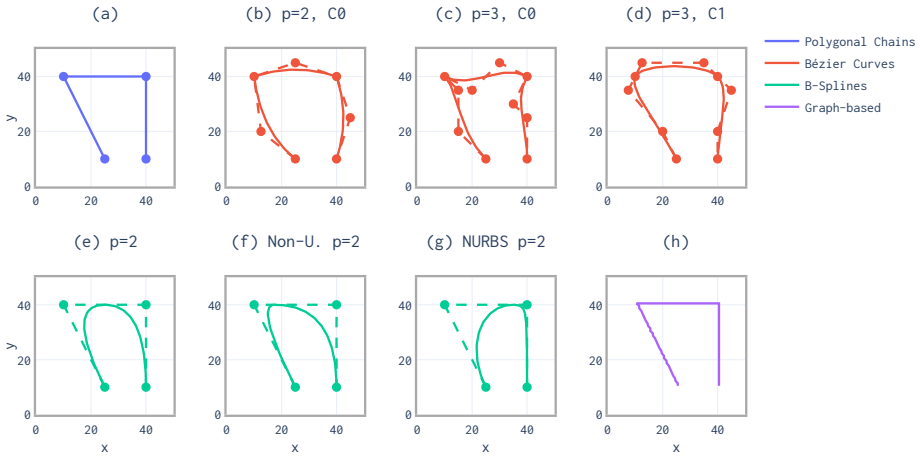

Fig. 1. Visualizations of selected path representations: Polygonal Chains (a), Composite Bézier Curves (b-d), B-Splines (e-g), and Graph-based (h). The additioinal markers connected by dashed lines indicate the control nodes of the Bézier curves and nodes of the B-Splines, respectively.

Polygonal Chains are the simplest path representation. They consist of N control points $[P_0, \ldots, P_{N-1}]$, connected by linear segments, as visualized in Figure 1 (a). While this representation does not feature any additional degree of freedom, its simplicity results in a very low computational complexity. The polygonal chain, and all following path representations, are defined over the parameter $u \in [0, 1]$. For this representation, each linear segment gets a unit interval of u assigned corresponding to the index of the first vertex. This results in the following mathematical formulation:

$$P_{Polygonal}(u) = P_i + \hat{u}\frac{P_{i+1} - P_i}{\|P_{i+1} - P_i\|} \tag{1}$$

with $i = \lfloor u(N-1) \rfloor$ (2) and $\hat{u} = u(N-1) \bmod 1$ (3)

Composite Bézier Curves consist of individual Bézier curves, each of degree p, with a C^0 continuity constraint. This constraint requires that the last point of one Bézier curve coincides with the starting point of the subsequent curve. Each segment is defined by $p+1$ control points as seen in Figure 1: (b-d) $[P_0^i, \ldots, P_p^i]$, where i is the index of the segment within the whole path. These segments can be expressed as

$$B^i(\hat{u}) = \sum_{j=0}^{p} \binom{p}{j} (1-\hat{u})^{p-j} \hat{u}^j P_j^i \tag{4}$$ and $$P_{Bézier}(u) = B^i(\hat{u}), \tag{5}$$

where i and $\hat{u}$ are based on the same parameterization method used for the polygonal chains and can be computed according to Equations 2 and 3. Additional continuity constraints can be established, but in this work, they are restricted to C^1 (velocity continuity). Its effect on the connections of the individual Bézier curves is clearly visible in Figure 1 (c) and (d). Stricter continuity constraints lead to a cascading loss of control over subsequent control points. Equations 6 and 7 for C^0 and C^1 continuity show how these constraints already affect the degree of freedom.

$$P_0^{i+1} = P_p^i \tag{6}$$ $$P_1^{i+1} = 2P_p^i - P_{p-1}^i \tag{7}$$

B-Spline paths are a type of spline represented as a linear combination of basis functions, each of order n and degree $p = n - 1$. The path is parameterized over u and the values $[u_0, u_1, \ldots, u_m]$ at which the polynomials meet are called knots. These knots are sorted in non-decreasing order. Given that $u \in [0, 1]$, a unique spline representation can be constructed as

$$P_{B-spline}(u) = \sum_j P_j B_{j,n}(u), \tag{8}$$

where P_j are the control points. The individual B-splines $B_{j,n}(u)$ can be constructed recursively, starting from order one

$$WB_{j,0}(u) := \begin{cases} 1 & if\ u_j \le u \le u_{j+1} \\ 0 & otherwise \end{cases} \tag{9}$$

$$B_{j,k+1}(u) := w_{j,k}(u) B_{j,k}(u) + [1 - w_{j+1,k}(u)] B_{j+1,k}(u) \tag{10}$$

with

$$w_{j,k} := \begin{cases} \frac{u-u_i}{u_{j+k}-u_i}, & u_{j+k} \neq u_j \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

In this paper, a B-spline has equidistant knots and is uniquely defined by the control points. If this restriction is not enforced, they are called non-uniform (N-U) B-splines and feature an additional degree of freedom. Further, they can also be rational (R). In this case, they are called NURBS for non-uniform rational B-splines. In addition to the control points and the knot vector, they feature a weight w for each control point. A higher weight pulls the curve towards the respective control point, as showcased in Figure 1 (h). In the case of the weights being all 1, NURBS generalize to non-uniform B-splines. A NURBS of order n or degree $p = n - 1$ can be defined by

$$P_{NURBS}(u) = \sum_j P_j R_{j,n}(u) \qquad (12) \qquad R_{j,n}(u) = \frac{B_{j,n}(u)w_j}{\sum_k B_{k,n}(u)w_k} \qquad (13)$$

where P_j are the control points, $R_{j,n}$ the rational basis functions, and $B_{j,n}$ the B-splines defined above.

Graph-based paths are implemented to serve as a comparison to traditional coverage path planning. They work on a regular grid with a von Neumann neighborhood topology. A path is encoded by a sequence of cell indices. The resulting path can be interpreted as a polygonal chain, with the center points of the visited cells being the control points.

2.2 Objectives

As motivated in the introduction, the main goal is to optimize paths that aggregate as much as possible of the value function. However, to be usable in the real world, the paths need to be efficient. This includes minimizing the length and ensuring smoothness, as fewer turns and sharp corners lead to more efficient traversal. These three objectives are formalized below.

Length is a crucial property of every path. The operational time of robots is often limited by their respective battery capacity, especially for UAVs. Therefore, the optimized paths should be short while achieving high coverage. The length of a path is defined as

$$f_1 = \int_0^1 |P'(u)| du, \qquad (14)$$

but it is solved via numerical integration or by summing the lengths of the linear segments in the case of polygonal chains and the graph-based approach.

Smoothness of a path determines how well a robot can traverse it. While UAVs can maneuver quite well, quick directional changes still cost time and increase energy consumption. For differentiable paths, the smoothness objective is based on the squared arc length derivative of the curvature. It is a physics-motivated measure to reduce jerk [17], which has been applied for optimizing smooth paths before [2]. The computation is shown in Equation 15 with $K(u)$

being the curvature for a given path $P(u)$. For paths that are not differentiable because they consist of $n+1$ linear segments, the smoothness is computed as the average absolute turning angles α_i between segment i and $i+1$ as shown in Equation 16.

$$f_2 = \int_0^1 \frac{K'(u)^2}{\sqrt{1+P'(u)^2}} \tag{15}$$

$$f_2 = \frac{\sum_{i=1}^{n} \alpha_i}{n} \tag{16}$$

Weighted Coverage is determined by the area visible while traversing a path, combined with the values of the value-function A. At each sampled position along the path, a square portion of the area can be seen, resulting in the matrix $V \in \mathbb{R}^{m \times n}$, as described in Equation 17. The proportion of the area covered by the evaluated path is calculated by dividing the element-wise multiplication of A and V by the total sum of A. To formulate this as a minimization problem, the fraction of the area that has not been covered is used as the objective, as shown in Equation 18.

$$V_{i,j} = \begin{cases} 1 & \text{if } (i,j)^T \text{ is seen} \\ 0 & \text{else.} \end{cases} \tag{17}$$

$$f_3 = 1 - \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{i,j}V_{i,j}}{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{i,j}} \tag{18}$$

2.3 No-Fly Zone Constraint

Compliance with NFZs is enforced by a single inequality constraint $g \leq 0$, where g represents the total length of the path sections that run within at least one NFZ. This results in the following equation:

$$g = \sum_i \int_0^1 |f(u, NFZ_i)| du \tag{19}$$

$$f(u, NFZ) = \begin{cases} P'(u) & \text{if } P(u) \in NFZ \\ 0 & \text{else} \end{cases} \tag{20}$$

3 Algorithm

The goal is to minimize the three objectives described in subsection 2.2 by optimizing the coverage paths. The NSGA-II algorithm [9] is employed, as it is a simple-to-use algorithm capable of generating solutions for three objectives. A binary tournament selector chooses the parents from the current population P, which are used to create the children C with $|P| = |C| = n_{pop}$. The encoding, custom initialization, crossover, and mutation operators are explained in the following sections. The optimization was implemented using pymoo [3].

3.1 Encodings

All representations, except for the graph-based one, are encoded as one-dimensional arrays of floating-point values. However, the number of decision variables differs among them. Therefore, the representations must be compared in terms of the dimension of the search space, given the number of waypoints N and the degree p or order n of the curves used. The resulting formulas are derived below and summarized in Table 1.

Polygonal chains require storing only a control point per waypoint. As the experiments are limited to two dimensions, the total number of decision variables is $2N$. A Bézier curve requires $p+1$ control points. When having $N-1$ curves, the number of decision variables totals to $2(N-1)(p+1)$. However, additional constraints make some of this information redundant. Due to the C^0 constraint, the first control point of all but the first curve are duplicates, reducing the number of variables to $2(N-1)p+2$. Similarly, the C^1 constraint makes the second control point of each Bézier curve redundant, as it must be symmetric to the second-to-last control point of the previous segment. Consequently, the number of values required to represent the path is $2(N-1)(p-1)+4$. For a cardinal B-spline, the curve is defined solely by its control points. Therefore, the number of decision variables is the same as for polygonal chains, namely $2N$. For non-cardinal B-splines, the knot vector also needs to be stored. It has $N+p+1$ knots, but the first and last $p+1$ knots are zero or one, respectively, to ensure the path starts and ends at the first and last waypoint. Therefore, the total number of decision variables is $3N-p-1$. NURBS extend B-splines by adding a weight vector, contributing N additional values, resulting in a total of $4N-p-1$. For graph-based paths, no simple equation can be provided, as it depends on the grid size and the length of the path.

Table 1. Comparison of the required decision variables. N is the number of waypoints and n and p are the order and degree, respectively.

Representation	Parameters	Constraints	Decision Variables
Polygonal Chain	N		$2N$
Bézier curves	N, p	C^0	$2(N-1)p+2$
		C^1	$2(N-1)(p-1)+4$
B-spline	N, p		$2N$
Non-Uniform B-spline	N, p		$3N-p-1$
NURBS	N, p		$N-p-1$

3.2 Initialization

All paths start from the same predetermined position. Additional points are sampled uniformly within the area. From the start point, these points are connected using a greedy approach, selecting the closest neighbor that has not yet

been included in the path. While polygonal chains can be directly instantiated from these ordered waypoints, representations with higher degrees of freedom are not uniquely defined. For Bézier curves, the additional control nodes are placed evenly along the lines connecting the waypoints. For B-splines, the waypoints are used directly as control nodes, with a uniform knot vector and equal weights of one. Lastly, for the graph representation, the Bresenham line algorithm is used to connect the waypoints on the grid.

3.3 Crossover

The crossover operator is implemented as a one-point crossover. A random waypoint index is selected from the shorter path, at which both paths are crossed. This is a simple implementation, as the focus lies on the representations. However, a more sophisticated version is possible and has already been used in other works. Instead of selecting a random waypoint to cross, an intersection between the two parent paths could be used. This results in more realistic offspring, as no new connections need to be made at the crossover point. The drawback, however, is that computing the intersection point can be computationally expensive, especially for B-spline-based representations.

3.4 Mutation

In the case of mutation, one of three independent operators is applied, each depicted schematically in Figure 2 and explained below. The first operator modifies either a waypoint or any other value of the respective representation. The other two operators add or delete a waypoint, allowing for the creation of shorter or longer paths. Since this paper aims to compare the representations themselves, the individual operators are implemented as consistently as possible across the representations. Though, different representations may benefit from individually tailored variation operators. However, such a comparison is not possible within the scope of this work.

The modification operator randomly selects a waypoint and translates it based on a randomly sampled vector $\vec{d}$, with the magnitude sampled from a normal distribution with mean μ and standard deviation σ and the angle from a uniform distribution. In the case of composite Bézier paths, this also applies to the additional control nodes. For representations based on B-splines, instead of modifying a control node, the additional values can be modified with equal chances: either a knot vector is replaced with a new value sampled from $\mathcal{U}_{(0.0,1.0)}$, or a weight is replaced with a new one sampled from $\mathcal{U}_{(0.0,2.0)}$. Implementing the same behavior for the graph-based representation is challenging, as the notion of a global waypoint is missing due to the underlying grid. Therefore, the two neighboring global waypoints are those with a distance on the path of approximately $\|\vec{d}\|$ to the selected point. For the mutation that adds a new waypoint, two subsequent waypoints are first selected. A new point is uniformly sampled from a circular area, with the center being the middle of the connecting line of the two selected waypoints and the diameter being the distance between them.

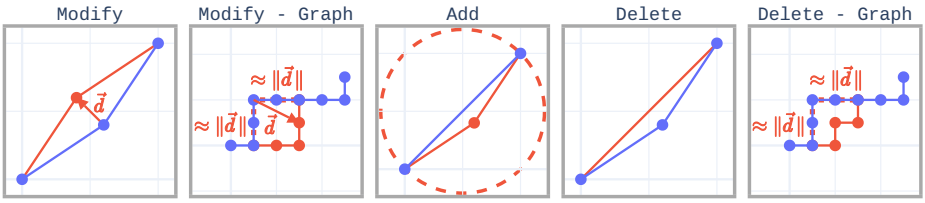

Fig. 2. Examples for the mutation operators, with the original path in blue and the mutated version in red.

Again, the graph-based representation requires an exception for the same reason as before: there is no notion of two neighboring global waypoints. As a result, the addition boils down to the modification mutation. Lastly, the delete mutation removes a randomly selected waypoint, except the starting point. For the graph-based approach, the same strategy as for the modify mutation is used: the two points between which the path is replaced with the shortest path are found by traversing on the path by a distance $\|\vec{d}\|$ in each direction from the selected point.

4 Experiments

A total of 14 different representations are tested: polygonal chains, quadratic composite Bézier curves with C^0 continuity, cubic composite Bézier curves with C^0 or C^1 continuity, quadratic or cubic B-splines, non-uniform B-splines or NURBS, and lastly the graph-based approach with cell sizes of 1.0, 2.0, 4.0, or 5.0. Larger cell sizes are not tested, as the view size is set to 5.0. These representations are tested on six different scenarios depicted in Figure 3. The scenarios vary in the complexity of the value function, ranging from a uniform distribution and sine and cosine functions, to a realistic example in scenario six, where a road-like structure is highlighted in the middle along with regions of special interest. Each scenario can be tested with and without the NFZs, resulting in 12 different test cases.

For initialization, all paths start in the lower-left corner of the area, and 49 additional points are sampled for each individual in the starting population. A population size of 100 is used, with a crossover probability of 0.9 and a mutation probability of 0.75. The parameters μ and σ, which determine the magnitude of the shift for the modify mutation, are set to 10 and 2.5, respectively. These values are based on preliminary experiments that are outside the scope of this paper but should be examined regarding their impact across the different path representations in more detail in future research through ablation studies. Each optimization terminates after 100,000 function evaluations. Finally, 31 runs are computed for each combination, resulting in 5,208 total runs. Moreover, since comparing all 14 different path representations at once is challenging, the best parameter configuration of each variant is determined first before comparing them against each other.

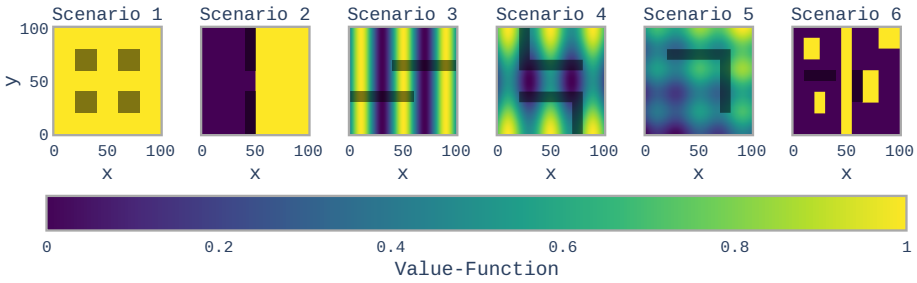

Fig. 3. The six used scenarios with their respective value functions. The shaded areas mark no-fly zones.

A visual data analysis based on the optimized fronts, HV, GD+, and IGD+ is used to determine the best parameters. Since no true Pareto front is known for this problem, nor can it be computed easily, it is approximated by combining all runs for each scenario and identifying the non-dominated front. However, there is no guarantee that it is well distributed, which is a requirement for IGD and its variants. As a result, the findings based on those metrics need to be interpreted with caution.

5 Evaluations

Parameter Comparison: The first row of Figure 4 shows two plots of the HV over the generations for the third scenario, with and without NFZs, comparing the different composite Bézier curve representations. Notably, the representation based on cubic Bézier curves with the C^1 constraint appears to perform the best. This observation holds for most other scenarios as well. The likely reason is that it features smooth curves between segments and a higher degree of freedom, resulting in higher coverage with the same number of segments.

When inspecting the GD+ and IGD+ metrics in subplot (c) and (d), no clear favorite emerges among the different B-spline variants. However, when looking at the HV over the generations, normal B-splines perform best in all 12 tested scenarios. An example is shown in subplot (e) of Figure 4. Out of the 12 scenarios, quadratic B-splines perform best in seven, while the remaining five are draws. This difference is due to quadratic B-splines generating longer paths with good coverage. A corresponding 2D Pareto front, that shows this difference, is visualized in plot (f) of Figure 4.

Lastly, the graph-based paths show similar performance across the HV, GD+, and IGD+ metrics. For this reason and due to space constraints, these illustrations are not included. A slight trend is visible in the distribution of the resulting fitness values. A smaller cell size leads, as expected, to a smoother path, while a larger cell size results in better coverage per path length. This effect is especially noticeable for solutions with higher coverage. The respective plots are illustrated

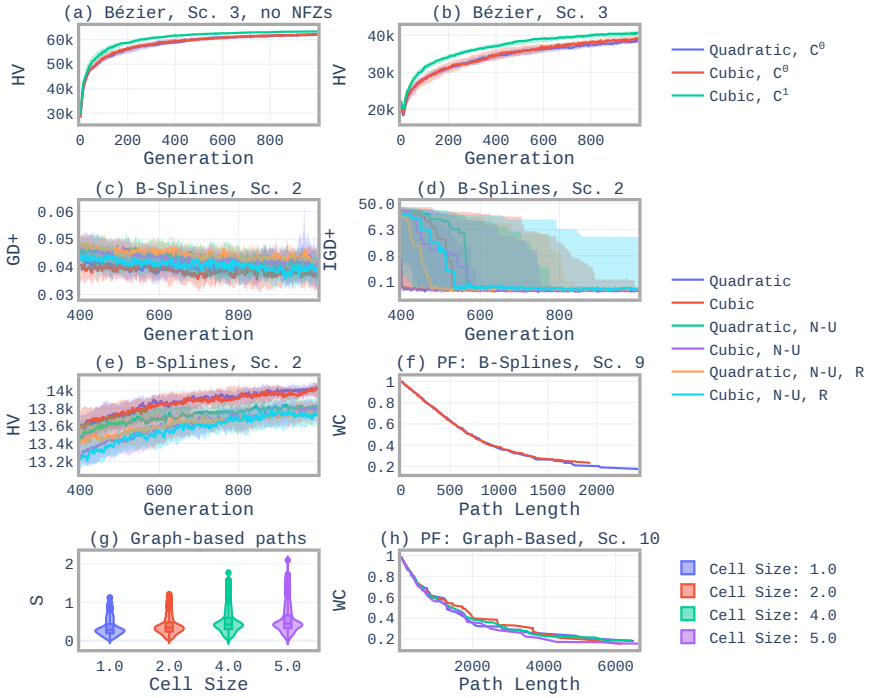

Fig. 4. Selected plots for the parameter comparison of the different path representations, with the shaded area marking values between the 25th and 75th percentile. (HV: Hypervolume, WC: Weighted Coverage, S: Smoothness, Sc.: Scenario, PF: Combined and connected Pareto Front)

in Figure 4 (g) and (h). Since the main goal of the optimization is to achieve good coverage, the largest cell size is selected for further comparisons with the other path representations. Overall, polygonal chains, composite cubic Bézier curves with a C^1 constraint, quadratic B-splines, and the graph-based approach with a cell size of 5.0 are selected for comparison in the next section.

Representation Comparison: Since the smoothness has been calculated differently, the remaining two objectives are used to compute a comparable HV. Additionally, only solutions that pass a certain smoothness threshold are considered. Firstly, polygonal chains and composite Bézier curves perform similarly. However, in most scenarios, the Bézier curves tend to perform slightly better. The IGD+ metric shows this difference, for example in the fourth scenario, depicted in plot (d) of Figure 6. Additionally, polygonal chains are inherently less smooth compared to composite Bézier curves with a C^1 continuity, as seen in the direct comparison between plots (e) and (f) of the same figure. Yet, polygonal chains have a substantial advantage in computational effort, as shown in Figure 5. When judging B-splines solely based on the HV, it is the worst representation. Not only the HV increases slowly, but also it does not always reach the

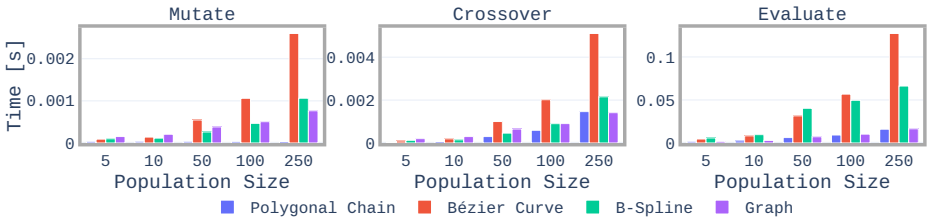

Fig. 5. Average runtime in seconds for the evaluate function and the mutation and crossover operator over the population size (Computed as single tasks with 31 CPU cores per task and 2 GB of RAM per core on a single node of a Slurm cluster [16] running an AMD Epyc 7543 with no additional workload on the same node)

same level as the other representations. The reason becomes clear when looking at the Pareto front itself in Figure 6 (b): B-splines actually find a better set of solutions for the most part but do not cover the entire front. Their advantage, however, is seen in the GD+ and IGD+ metrics in plots (c) and (d), respectively. Runtime-wise, B-splines outperform Bézier curves but lose against the graph-based and polygonal chain representations. The graph-based approach has its distinct advantages and disadvantages: As visible in plot (a) of Figure 6, the HV initially increases the fastest but then flattens out or even decreases. Plot (b) of the same figure shows the reasons: the approach is excellent in diversity, covering the whole front and generating solutions not found by other representations, but the convergence is comparatively poor, which is strongly reflected in the GD+ metric. The bottom row of Figure 6 depicts one path per representation that has the best coverage while being between 950 and 1050 units long. All representations can prioritize the yellow, more important areas. Notably, the polygonal chains and B-splines require fewer crossing paths in the less important areas. Apart from the inherently better smoothness of the Bézier and B-spline representations, graph-based paths are the only representations that sometimes resemble how a human would plan a path.

6 Conclusion and Future Work

This paper proposed and compared four different mathematical representations for free-form paths used to optimize a weighted coverage path planning problem using a multi-objective algorithm. The representations evaluated were polygonal chains, composite Bézier curves, B-splines, and a graph-based approach. In the first step, the best parameters for each representation were determined: composite cubic Bézier curves with a C^1 continuity, quadratic cardinal B-splines, and a cell size equal to the visible area of a UAV for the graph-based approach. All representations were able to optimize solutions even for complex scenarios and find suitable trade-offs. Nonetheless, differences were observed that make some approaches more or less suitable for certain scenarios: **Polygonal chains** are

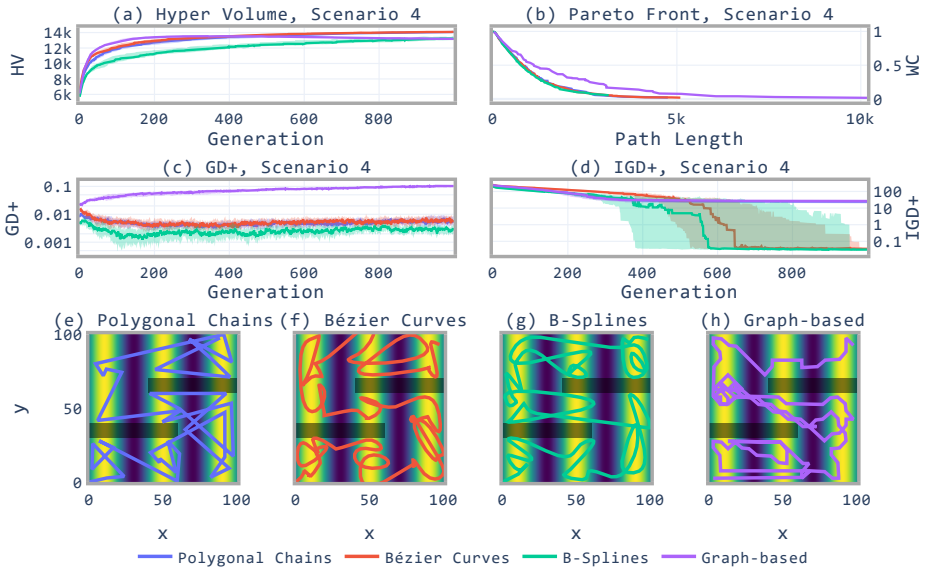

Fig. 6. The three metrics HV, GD+, and IGD+ in (a), (c), and (d), respectively, plus the combined Pareto front for each representation of the fourth scenario in (b) with WC being the weighted coverage. The shaded area marks values between the 25th and 75th percentile. (e)-(h) in the last row show solutions with a length between 950 and 1050 with the best respective coverage.

substantially fast and feature good performance but are inherently not smooth. **Composite cubic Bézier curves with C^1 continuity** often slightly outperform polygonal chains and are inherently smooth, but at a notable increase in computational cost. **Cardinal B-splines** show good convergence but lack diversity, providing the best solutions for most of the Pareto front while missing the extreme parts. Runtime-wise, they are between polygonal chains and Bézier curves. The **Graph-Based approach** behaves contrary to B-splines, featuring good diversity but poor convergence. This leads to an initially fast increase in the HV that flattens out quickly, resulting in final values lower than those of the other representations. However, the mathematical representation is only one part of the optimization. Mutation and crossover operators can also significantly impact the performance of the individual representations, which need to be evaluated in future research. Additionally, in this study, exclusively NSGA-II was employed; other algorithms, such as indicator-based or decomposition-based approaches, should also be investigated for their applicability.

Acknowledgments. This research was partially funded by the German Research Foundation (DFG) under project number 502167710.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Batista, V.R., Zampirolli, F.A.: Optimising Robotic Pool-Cleaning with a Genetic Algorithm. Journal of Intelligent & Robotic Systems **95**(2), 443–458 (Aug 2019). https://doi.org/10.1007/s10846-018-0953-y
2. Berglund, T., Brodnik, A., Jonsson, H., Staffanson, M., Soderkvist, I.: Planning Smooth and Obstacle-Avoiding B-Spline Paths for Autonomous Mining Vehicles. IEEE Transactions on Automation Science and Engineering **7**(1), 167–172 (Jan 2010). https://doi.org/10.1109/TASE.2009.2015886
3. Blank, J., Deb, K.: Pymoo: Multi-Objective Optimization in Python. IEEE Access **8**, 89497–89509 (2020). https://doi.org/10.1109/ACCESS.2020.2990567
4. Bostelmann-Arp, L., Steup, C., Mostaghim, S.: Linking Field Decomposition and Coverage Path Planning: A Coevolution Approach. In: 2023 IEEE Conference on Artificial Intelligence (CAI). pp. 294–295. IEEE, Santa Clara, CA, USA (Jun 2023). https://doi.org/10.1109/CAI54212.2023.00131
5. Bostelmann-Arp, L., Steup, C., Mostaghim, S.: Multi-Objective Seed Curve Optimization for Coverage Path Planning in Precision Farming. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 1312–1320. ACM, Lisbon Portugal (Jul 2023). https://doi.org/10.1145/3583131.3590490
6. Calamoneri, T., Coro, F., Mancini, S.: A Realistic Model to Support Rescue Operations After an Earthquake via UAVs. IEEE Access **10**, 6109–6125 (2022). https://doi.org/10.1109/ACCESS.2022.3141216
7. Chen, H., Xie, H., Sun, L., Shang, T.: Research on Tractor Optimal Obstacle Avoidance Path Planning for Improving Navigation Accuracy and Avoiding Land Waste. Agriculture **13**(5), 934 (Apr 2023). https://doi.org/10.3390/agriculture13050934
8. Dai, R., Fotedar, S., Radmanesh, M., Kumar, M.: Quality-aware UAV coverage and path planning in geometrically complex environments. Ad Hoc Networks **73**, 95–105 (May 2018). https://doi.org/10.1016/j.adhoc.2018.02.008
9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
10. Ellefsen, K., Lepikson, H., Albiez, J.: Multiobjective coverage path planning: Enabling automated inspection of complex, real-world structures. Applied Soft Computing **61**, 264–282 (Dec 2017). https://doi.org/10.1016/j.asoc.2017.07.051
11. Elshamli, A., Abdullah, H., Areibi, S.: Genetic algorithm for dynamic path planning. In: Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No.04CH37513). pp. 677–680. IEEE, Niagara Falls, Ont., Canada (2004). https://doi.org/10.1109/CCECE.2004.1345203
12. Flores-Caballero, G., Rodriguez-Molina, A., Aldape-Perez, M., Villarreal-Cervantes, M.G.: Optimized Path-Planning in Continuous Spaces for Unmanned Aerial Vehicles Using Meta-Heuristics. IEEE Access **8**, 176774–176788 (2020). https://doi.org/10.1109/ACCESS.2020.3026666
13. Hameed, I.A., Bochtis, D., Sørensen, C.A.: An Optimized Field Coverage Planning Approach for Navigation of Agricultural Robots in Fields Involving Obstacle Areas. International Journal of Advanced Robotic Systems **10**(5), 231 (May 2013). https://doi.org/10.5772/56248
14. Hu, C., Jin, Y.: Path Planning for Autonomous Systems Design: A Focus Genetic Algorithm for Complex Environments. Journal of Autonomous Vehicles and Systems **2**(4), 041001 (Oct 2022). https://doi.org/10.1115/1.4063013

15. I. A. Hameed, D. D. Bochtis, C. G. Sorensen: Driving Angle and Track Sequence Optimization for Operational Path Planning Using Genetic Algorithms. Applied Engineering in Agriculture **27**(6), 1077–1086 (2011). https://doi.org/10.13031/2013.40615
16. Jette, M.A., Wickberg, T.: Architecture of the Slurm Workload Manager. In: Klusáček, D., Corbalán, J., Rodrigo, G.P. (eds.) Job Scheduling Strategies for Parallel Processing, vol. 14283, pp. 3–23. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-43943-8_1, series Title: Lecture Notes in Computer Science
17. Kanayama, Y., Hartman, B.: Smooth local path planning for autonomous vehicles. In: Proceedings, 1989 International Conference on Robotics and Automation. pp. 1265–1270. IEEE Comput. Soc. Press, Scottsdale, AZ, USA (1989). https://doi.org/10.1109/ROBOT.1989.100154
18. Leng, S., Sun, H.: UAV Path Planning in 3D Complex Environments Using Genetic Algorithms. In: 2021 33rd Chinese Control and Decision Conference (CCDC). pp. 1324–1330. IEEE, Kunming, China (May 2021). https://doi.org/10.1109/CCDC52312.2021.9601765
19. Mahamat Pierre, D., Zakaria, N.: Genetic Algorithm Approach to Path Planning for Intelligent Camera Control for Scientific Visualization. In: Zain, J.M., Wan Mohd, W.M.B., El-Qawasmeh, E. (eds.) Software Engineering and Computer Systems, vol. 180, pp. 205–213. Springer Berlin Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22191-0_18, series Title: Communications in Computer and Information Science
20. Ou, J., Hong, S.H., Ziehl, P., Wang, Y.: GPU-based Global Path Planning Using Genetic Algorithm with Near Corner Initialization. Journal of Intelligent & Robotic Systems **104**(2), 34 (Feb 2022). https://doi.org/10.1007/s10846-022-01576-6
21. Roberge, V., Tarbouchi, M., Labonte, G.: Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning. IEEE Transactions on Industrial Informatics **9**(1), 132–141 (Feb 2013). https://doi.org/10.1109/TII.2012.2198665
22. Sadek, M.G., Mohamed, A.E., El-Garhy, A.M.: Augmenting Multi-Objective Genetic Algorithm and Dynamic Programming for Online Coverage Path Planning. In: 2018 13th International Conference on Computer Engineering and Systems (ICCES). pp. 475–480. IEEE, Cairo, Egypt (Dec 2018). https://doi.org/10.1109/ICCES.2018.8639412
23. Tsouros, D.C., Bibi, S., Sarigiannidis, P.G.: A Review on UAV-Based Applications for Precision Agriculture. Information **10**(11), 349 (Nov 2019). https://doi.org/10.3390/info10110349
24. Xue, Y.: Mobile Robot Path Planning with a Non-Dominated Sorting Genetic Algorithm. Applied Sciences **8**(11), 2253 (Nov 2018). https://doi.org/10.3390/app8112253
25. Yao, H., Qin, R., Chen, X.: Unmanned Aerial Vehicle for Remote Sensing Applications—A Review. Remote Sensing **11**(12), 1443 (Jun 2019). https://doi.org/10.3390/rs11121443