

Learning Obstacle-Avoiding Lattice Paths using Swarm Heuristics: Exploring the Bijection to Ordered Trees

Victor Parque

Department of Modern Mechanical Engineering, Waseda University
3-4-1 Okubu Shinjuku Tokyo, 169-8555, Japan
parque@aoni.waseda.jp

Abstract—Lattice paths are functional entities that model efficient navigation in discrete/grid maps. This paper presents a new scheme to generate collision-free lattice paths with utmost efficiency using the bijective property to rooted ordered trees, rendering a one-dimensional search problem. Our computational studies using ten state-of-the-art and relevant nature-inspired swarm heuristics in navigation scenarios with obstacles with convex and non-convex geometry show the practical feasibility and efficiency in rendering collision-free lattice paths. We believe our scheme may find use in devising fast algorithms for planning and combinatorial optimization in discrete maps.

Index Terms—path planning, lattice paths, ordered trees, enumeration, combinatorial objects, Catalan numbers

I. INTRODUCTION

Trees allow to model hierarchical dependencies in combinatorial problems ubiquitously [1]. Among the existing class of trees, the family of ordered trees are well-known to have a bijection to Catalan numbers, binary trees with n external nodes, trees with n nodes, legal sequences of n pairs of parentheses, triangulated n -gons, and lattice paths [2].

How to generate ordered trees has attracted the favorable attention in the discrete mathematics community. Among the existing approaches, integer sequences are known to generate ordered trees with n vertices and k leaves in $O(n - k)$ time [3]. Also, it is possible to traverse the *genealogy of trees* with at most n vertices in $O(n)$ space and $O(1)$ time per tree in average [4]. The *genealogy of trees* is in essence the tree of trees, and offers a systematic means to render all families of trees. For instance, it is possible to enumerate ordered trees with exactly n vertices and k leaves in $O(1)$ time in the worst case [5]. Most of the above-mentioned approaches basically extend the notion of reverse search [6], which generates objects through a graph (tree) whose edges model local and bounded operations on the objects, thus it becomes possible to generate entities (trees) by traversing the graph backwards by an adjacency expansion oracle. Authors have also explored the idea of complete generation of trees [7], and others have used distinct mechanisms such as parent arrays [8], [9], level sequences [10], structural constraints [11], binary trees [12], triangulations of convex polygons [13] and lattice enumerations [14].

In this paper we tackle the problem of planning obstacle-avoiding paths over grid maps by using gradient-free heuristics and a representation rendered from the bijection between lattice paths and the class of ordered trees. Generally speaking, path planning by heuristic algorithms are often based on A* [15], Genetic Algorithms [16], [17], Differential Evolution [18], [19], Heuristic Graph Search [20], Ant Colony Optimization [21], Particle Swarm Optimization [22] and Local Search [23], [24]. Basically, the path planning problem has been approached from the mutation, crossover and hybrid selection perspectives, in which paths are often represented by a set of points or control commands whose optimal configuration is to be found by the optimization heuristic. Thus, the problem scales with the number of points or commands used in the encoding.

On the other hand, we exploit the 1-1 correspondence between ordered trees and the space of lattice paths [9] to devise an heuristic to generate obstacle-avoiding lattice paths, whose unique benefit is to render a one-dimensional search problem, which is tractable by the state of the art search/optimization heuristics. The study of the path planning problem using the bijection to the ordered trees and its feasibility study by swarm optimization algorithms is the first proposed in the literature, to the best of our knowledge. Basically, we present a new approach to generate collision-free lattice paths in grid maps by using a bounded search space over the combinatorial encoding of ordered trees. Our contributions are summarized as follows:

- a recursive approach that generates obstacle-avoiding lattice paths over a one-dimensional search space.
- the computational studies using relevant nature-inspired swarm heuristics using diverse set of exploration-exploitation features in convex and non-convex navigation scenarios.

Our experiments demonstrated that Strategy Adaptation Differential Evolution (SADE) and the swarm algorithms showing the explorative features outperformed their counterparts with exploitative features. These observations suggest that the swarm-based algorithms using explicit exploration mechanisms offer potential merits to tackle the lattice path planning

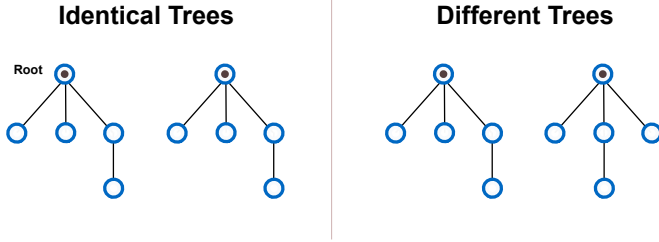


Fig. 1: Main notion of ordered trees

problem using the ordered tree representation effectively.

II. PROPOSED METHOD

In this section, we present the main concepts and algorithms involved in our proposed approach.

A. Encoding Mechanism

Due to the nature of the implicit order of leaves, two ordered trees are similar (different) if the order of leaves follow the same (different) order with respect to some convention, as the one from left to right as shown by Fig. 1. The nature of such order allows one to use encodings based on tree traversal such as the pre-order or post-order arrangements to identify nodes in the tree systematically. In this paper, we use the following tuple-based mechanism to represent an ordered tree:

$$t = (t_1, t_2, \dots, t_i, \dots, t_n), \quad (1)$$

where an ordered rooted tree has n nodes and t_i stands for the number of children of the i -th node of the tree in preorder traversal, thus

$$t_n = 0, \quad t_i \in [0, n-1]. \quad (2)$$

The above-mentioned representation in (1) has a 1-1 bijection to the family of lattice paths in a grid with $n \times n$ nodes as shown by the example of Fig. 2. In such case, t_i denotes the relative height in the grid.

Also the above tuple-based representation is inspired by the BCT representation of binary trees, in which B stands for branching, C stands for continuation, and T stands for terminal [25], [26].

- The BCT representation is renderable from the column-wise sum of elements below the diagonal of the adjacency matrix representing the tree, and
- The adjacency matrix is renderable from the BCT encoding by an $O(n)$ algorithm based on stacks [27].

Under the BCT approach, if ordered trees were represented by an adjacency matrix, and assuming tree nodes are labeled by a user-defined order, each element of the tuple t in (1) is equivalent to the column-wise sum of elements below the adjacency matrix representation.

To give an example of the 1-1 bijective relationship between ordered trees and lattice paths, Fig. 3 shows the node labels that define the traversal order in pre-order and their 1-1 parallel

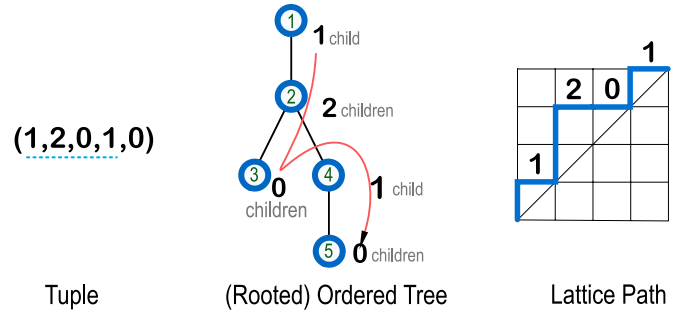


Fig. 2: Basic idea of the encoding mechanism. Each element of the tuple denotes the number of children of the ordered tree in preorder labeling of the nodes. Also, each element of the tuple denotes the relative height of the lattice path.

to lattice paths; and Fig. 4 shows examples of the encoding of diverse lattice paths of a square grid of $n = 5$.

Furthermore, due to the tuple t encodes the relative height of the node in the $n \times n$ lattice, the following holds:

$$\sum_{i=1}^n t_i = n - 1 \quad (3)$$

The above can also be derived from the following notion: since t_i is equivalent to the number of edges of the i -th node, then summing up all elements of the tuple t will render the total number of edges of the tree, that is $n - 1$.

B. Generating Lattice Paths

By using the tuple-based representation in (1), we propose a mechanism to generate arbitrary lattice paths which are above the diagonal. As such, it is possible to generate lattice paths by finding each element t_i of the tuple t for $i \in [n]$ by the following relationships:

$$t_i \sim \mathbf{U}\{L_i, U_i\} \quad i = 1, 2, \dots, n \quad (4)$$

$$L_i = 1 - \text{sgn}(t_{i-1} + S_{i-1} - 1) \quad (5)$$

$$S_i = S_{i-1} + t_{i-1} - 1 \quad (6)$$

$$U_i = U_{i-1} - t_{i-1} \quad (7)$$

where L_i and U_i are the lower and upper bound on t_i , respectively, such that $t_i \in [L_i, U_i]$, and $\text{sgn}(\cdot)$ denotes the signum function. Since the above mechanism is recursive in nature, we set the initial conditions $S_0 = 0$, $U_0 = n$, $t_0 = 1$. The variable S computes the accumulation of elements of the tuple. It is possible to eliminate the variable S , by which an equivalent expression to Eq. 7 is

$$L_i = 1 - \text{sgn}\left(\sum_{j=0}^{i-1} (t_j - 1)\right) \quad (8)$$

For ordered trees with n nodes, and considering the ordered nature of the tuple t and of Eq. 5 and Eq. 7, the following

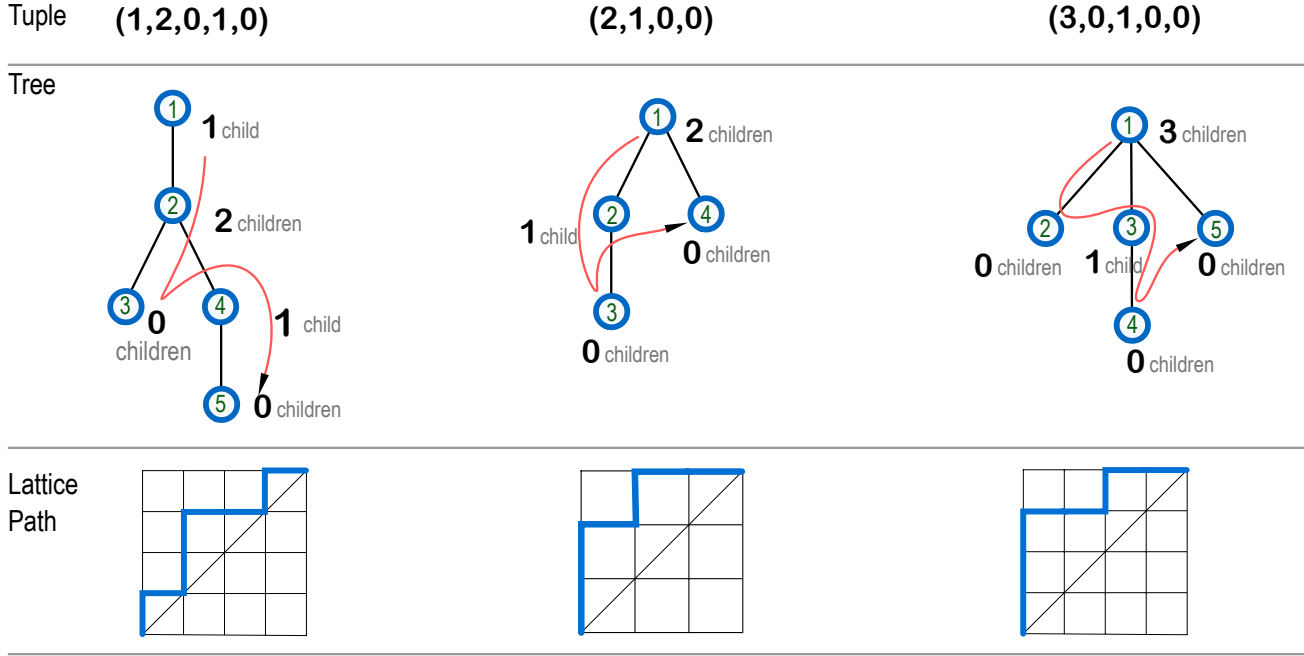


Fig. 3: Examples of the bijection between lattice paths and ordered trees. Elements of the tuple denoted above the picture denote the number of children of each node in the tree in preorder encoding. To represent the lattice path, the digits of the tuple encode the relative column height.

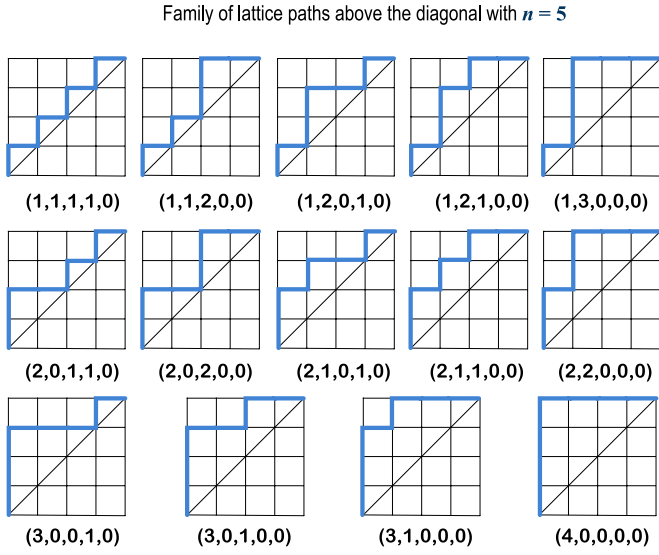


Fig. 4: The 1-1 correspondence with the lattice paths above the diagonal of a grid with $(n-1) \times (n-1)$ square cells for $n = 5$.

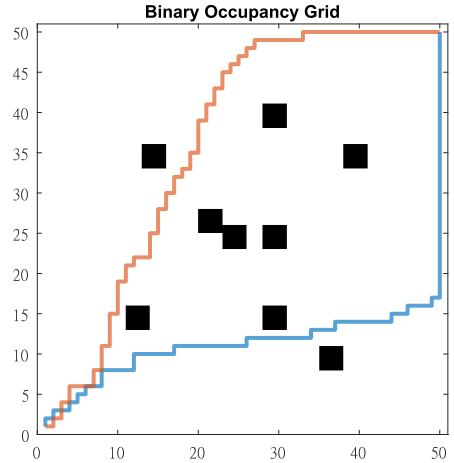


Fig. 5: Example of obstacle-avoiding lattice paths above and below the diagonal.

relations hold: $L_1 = 1$, $U_1 = n - 1$, and $L_n = U_n = 0$, thus $t_n = 0$, which aligns well with Eq. 2, and

$$t_1 \in [1, n - 1]. \quad (9)$$

C. Sampling Lattice Paths

Since the bounds for t_i are given as

$$t_i \in [L_i \dots U_i], \quad (10)$$

it becomes possible to compute the tuple t stochastically to realize the sampling mechanism of (4) as follows:

$$t_i = \left\lfloor L_i + \lambda(U_i - L_i) \right\rfloor, \quad (11)$$

$$\lambda = \mathbf{r} \cdot \left(\frac{x_i}{n} \right)^\alpha, \quad (12)$$

where λ is a normalization factor, \mathbf{r} is a random number with uniform distribution $\mathbf{U}[0, 1]$, α is curvature preference, x_i is the x-coordinate of the i -th node of the lattice path, for $i = 1, 2, \dots, n$. The range of $\alpha \in [0, \infty]$: small values of α (close to zero) denote L-shaped lattices paths (orthogonal to both x-axis and y-axis), and the large values of α denote lattice paths being close to the diagonal.

Computing (5) - (7) implies a recursive procedure with

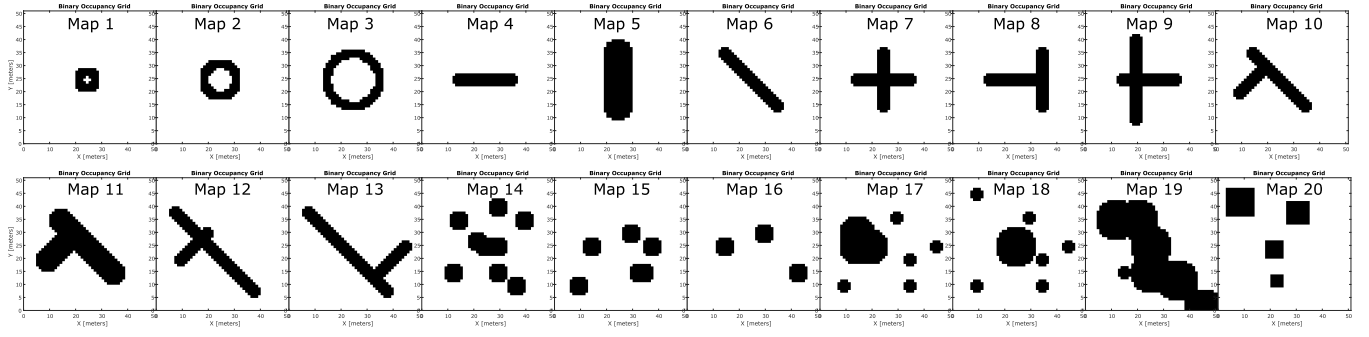


Fig. 6: Grid map environments used for evaluation.

Algorithm 1 Generate Lattice Path

```

1: function GENERATE PATH( $L, U, S, n, \alpha, \lambda, x, y$ )
2:   if  $n > 1$  then
3:     Compute  $t = \left\lfloor L + \lambda(U - L) \right\rfloor$ 
4:     if  $x = 1$  then
5:        $y = t$ 
6:     else
7:        $y = y + t$ 
8:     end if
9:      $O_{x,y} \leftarrow$  Get Occupancy at  $(x, y)$ 
10:    if  $O_{x,y} = \text{False}$  then
11:       $u \leftarrow U - t$ 
12:       $S \leftarrow S + t - 1$ 
13:       $l \leftarrow 1 - \text{sgn}(S)^\alpha$ 
14:       $\lambda = \mathbf{r} \cdot \left(\frac{x}{n}\right)$ 
15:       $n \leftarrow n - 1$ 
16:       $t' \leftarrow$  GENERATE PATH( $l, u, S, n, \alpha, \lambda, x + 1, y$ )
17:      return  $t \cup t'$ 
18:    else
19:      return  $\{\}$ 
20:    end if
21:  return  $\{\}$ 
22: end if
23: end function

```

$\mathcal{O}(n)$ space and $\mathcal{O}(1)$ time in average per path, realized by algorithm 1 generating lattice paths above the diagonal. Here, inputs are: the lower bound L of the i th element of the tuple t , the upper bound U of the i th element of the tuple t , the summation term S , the number of nodes n in the grid, the preference for curvature α , the normalization factor λ , the initial coordinates in the grid (x, y) . Note that line 9 of algorithm 1 checks the occupancy at coordinate (x, y) , thus it is possible to compute collision-free paths above the diagonal.

A lattice path starting at $(0, 0)$ and ending at the upper corner of the grid is generated by algorithm 1 with inputs $L = 1, U = n - 1, S = 0, \lambda = 0, x = 0, y = 0$ and the user-defined parameter α . Although algorithm 1 generates a path above the diagonal, it is possible to swap the coordinate x

by the coordinate y to generate paths below the diagonal. For instance, Fig. 5 shows two lattice paths, one above the diagonal (with $\alpha = 1.2$), and another below the diagonal (with $\alpha = 2.5$), implying that the parallel execution of algorithm 1 allows to compute multiple collision-free paths above and below the diagonal, for which the path topology depends on the one-dimensional parameter α . In Fig. 5, the origin is located at the bottom-left, while the destination is located at the top-right corner. In the following section, we evaluate the possibility of using gradient-free optimization heuristics to find optimal values of α for distinct navigation conditions.

III. COMPUTATIONAL EXPERIMENTS

In order to evaluate the feasibility of generating collision-free paths, we performed computational experiments comprising diverse grid environments. Our algorithm was implemented in Matlab 2020a and evaluations considered the environments with grid maps with 50×50 cells and obstacles comprising convex and non-convex geometry as shown by Fig. 6. Our computing environment was as follows Intel Core i7 @3.6GHz, 16 GB RAM. The time to generate lattice paths for a fixed α was in the order of 0.01 seconds, implying the fast performance for embedded platforms.

In order to show the performance frontiers of our approach, we used the following swarm heuristics to find optimal values of α : Particle Swarm Optimization (PSO) [28], Particle Swarm Optimization with Speciation (PSOSP) [29], Differential Particle Scheme (DPS) [30], Particle Swarm with Fitness Euclidean Ratio (PSOFER) [31], Differential Evolution with BEST/1/BIN mutation (DEBEST) [32], Differential Evolution with RAND/1/BIN mutation (DERAND) [32], Differential Evolution with Similarity-based Mutation Vector (DESIM) [33], Strategy Adaptation Differential Evolution (SADE) [34], Differential Evolution With Underestimation-Based Multimutation Strategy (UMSSADE) [35] Differential Evolution with Rank-based Mutation (RBDE) [36]. For simplicity and without loss of generality, we set to obtain obstacle-avoiding lattice paths above the diagonal with origin at $(0, 0)$ and destination at (n, n) . Our motivation of using the above algorithmic set is to include distinct mechanisms of selection pressure, multimodality and trade-off mechanisms in exploration-exploitation. For each scenario and each algorithm, the objective function F is the Euclidean distance of the path from the origin (bottom-

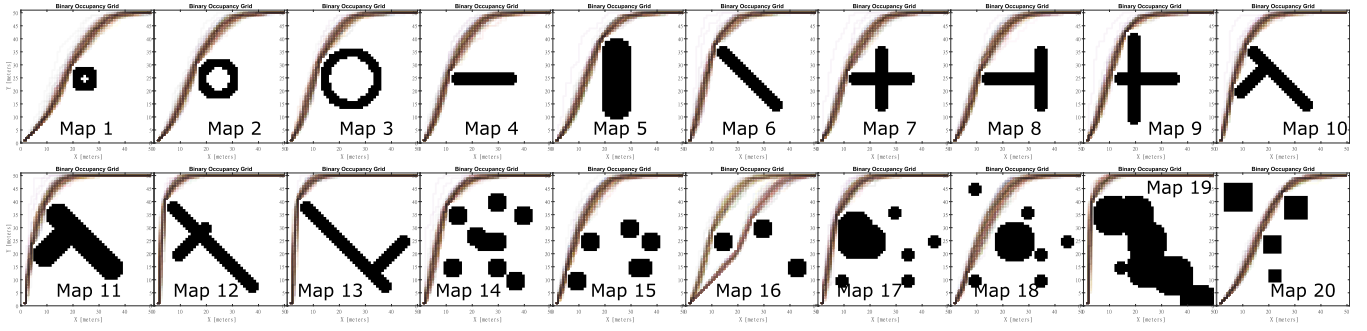


Fig. 7: Obtained paths in each grid map environment over 20 independent runs.

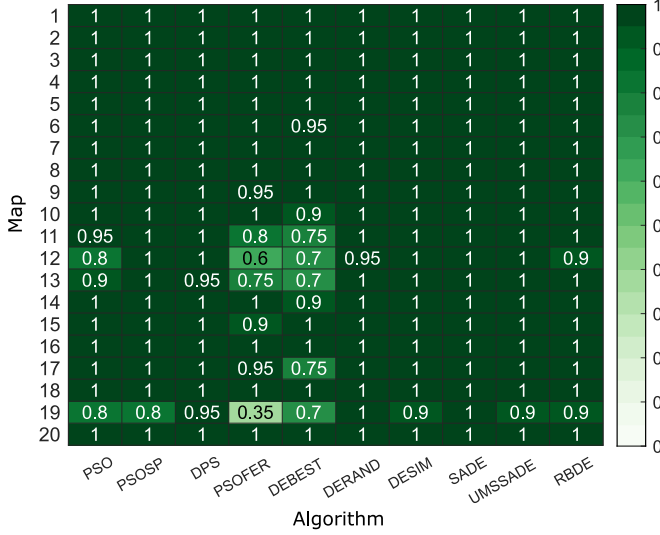


Fig. 8: Success ratio of each algorithm over 20 independent runs.

left) to the destination (top-right) in the 50×50 grid map. For evaluations, we used 20 independent runs with 1000 function evaluations at the maximum, which allows to compare the behaviour over independent random initializations under tight computational budgets. Parameters for PSO-based schemes involve $\omega = 0.7$, $c_1 = 2.05$, $c_2 = 2.05$, population size was set as 10. As for Differential Evolution algorithms, the crossover rate was $CR = 0.5$, the scaling factor $F = 0.7$. The coefficient β involved in the Whitley distribution scheme in Rank-Based Differential Evolution (RBDE) was set as $\beta = 2$. The fine tuning of the afore-mentioned variables is out of the scope of this study. Other parameters used the default settings in the respective references.

To show the overall performance of all evaluated algorithms, Fig. 7 shows the obtained obstacle-avoiding lattice paths by all algorithms in all environments and all independent runs. Here, lattice paths are rendered with a transparency factor, thus lattice paths with darker colors imply the most common navigable regions. Also, by looking at Fig. 7 one can note that it is possible to obtain multimodal lattice paths over independent runs such as the case of Map 6.

In order to show the effectiveness of each swarm optimization algorithm, Fig. 8 shows the ratio of success at finding

obstacle-avoiding lattice paths overall independent runs. For instance, by observing Fig. 8, the ratio of 1 (0.8) denotes that the algorithm was able to find obstacle-avoiding lattice paths in 100% (80%) of the independent runs. Among the studied optimization algorithms, we can observe from Fig. 8 that only SADE was able to generate obstacle-avoiding lattice paths in all independent runs, and that maps 11, 12, 13 and 19 were the most challenging environments for most algorithms. We can also observe that the heuristics with exploitation features such as PSOFR and DEBEST are outperformed by heuristics having exploration and diversity inducing mechanisms such as SADE, DERAND, DESIM and DPS. Furthermore, most algorithms were able to find obstacle-avoiding lattice paths in the first ten environments, implying that the convex/nonvex nature of the obstacles has no significant effect on the performance of the algorithm. On the other hand, maps with narrow passages such as maps 11, 12, 13 and 19 were challenging for most optimization algorithms.

To show the convergence characteristics of the studied algorithms, Fig. 9 shows the mean convergence behaviour over independent runs. By observing Fig. 7 and Fig. 9, we can note the feasibility to compute obstacle-avoiding lattice paths in challenging navigation scenarios with reasonable number of function evaluations. The noisy behaviour of some convergence figures at Fig. 9, e.g. at Map 12, 13, and 19 is due to some swarm optimization algorithms being unable to find obstacle-avoiding lattice paths in some independent runs (variability in the averaging over independent runs).

On the other hand, Fig. 10 shows the best (shortest) path obtained by each optimization algorithm over all independent runs, and Fig. 11 shows the lower bound of the convergence over independent runs. By observing Fig. 10 and Fig. 11, one can note that it is possible to obtain obstacle-avoiding lattice paths with similar topology, and that convergence occurs irrespective of the nonlinear stochastic algorithm over independent runs. It is also possible to study the statistical difference between the studied algorithms as shown by Fig. 12, which shows that explorative strategies are consistent in finding shortest lattice paths over all environments. The above-mentioned observations highlight the importance of exploration rather than exploitation when tackling the lattice path-finding problem. Investigating the fitness landscape for diverse geometries and large n is left to future work.

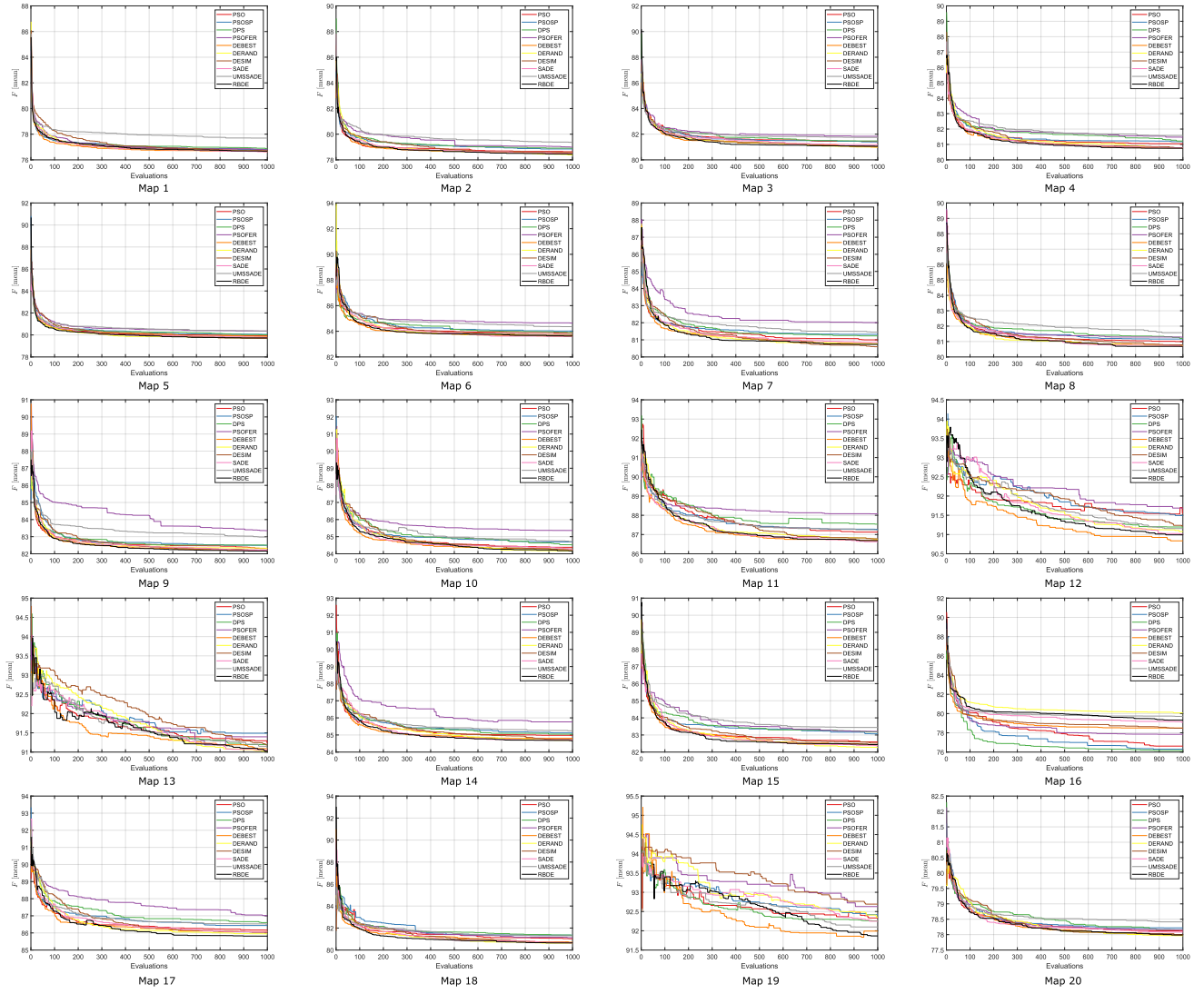


Fig. 9: Mean convergence of the evaluated algorithms over 20 independent runs.

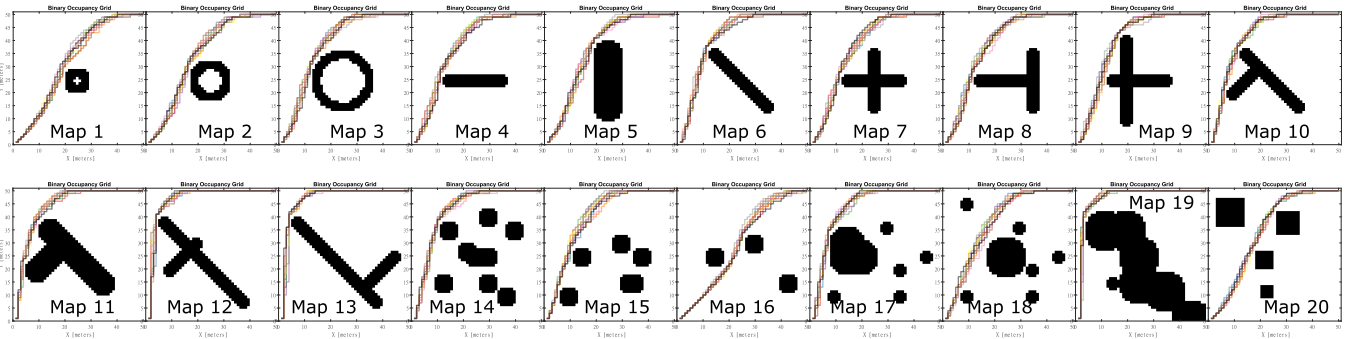


Fig. 10: Best obtained paths in each grid map environment over 20 independent runs.

The above-mentioned results show the feasibility and efficiency to generate collision-free lattice paths in grid maps. Due to one dimensional search problem and the 1-1 bijective property to ordered trees, our approach is potential to sample other combinatorial objects such as legal sequences of n pairs of parentheses, triangulated n -gons, and other combinatorial

objects based on catalan numbers. In future work, we aim at studying the smoothness considerations in paths [19], [37], the integration with trajectory tracking [38], the online adaptation and integration with other intelligent schemes such as Fuzzy Logic [39], the performance for very large n and their further applications in combinatorial optimization in Robotics

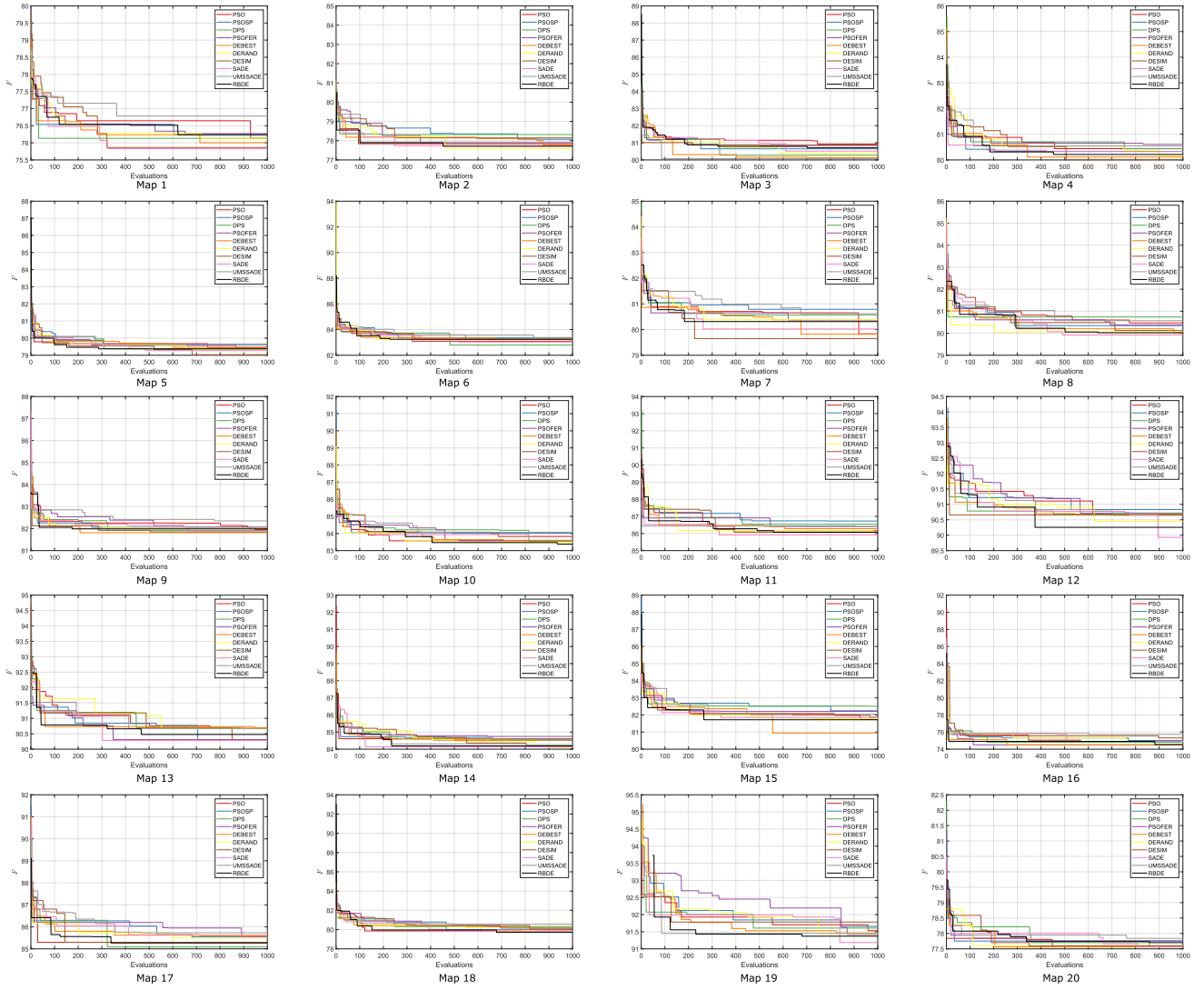


Fig. 11: Lower bound of the convergence of the evaluated algorithms over 20 independent runs.

and Operations Research. We believe the proposed approach may find its use in planning and combinatorial optimization problems.

IV. CONCLUSION

In this paper, we have proposed a new approach to generate obstacle avoiding lattice paths based on the 1-1 bijection to ordered trees, rendering a one dimensional optimization problem. Our computational studies using relevant swarm-based optimization heuristics has shown the feasibility to generate obstacle-avoiding lattice paths in challenging navigation scenarios consisting both of convex and non-convex obstacle geometries. Furthermore, our observations suggest that the explorative search strategies are effective over independent runs. In future work, we aim at studying further combinatorial optimization problems in Robotics and Operations Research; for instance the combinatorial problems involving binary trees with n external nodes, the legal sequences of n pairs of

parentheses in modularity formation, and the triangulated n -gons in folding problems.

REFERENCES

- [1] I. Aldana-Galván, J. Catana-Salazar, J. D.-B. nez, F. Duque, R. Fabila-Monroy, M. Heredia, A. Ramírez-Vigueras, and J. Urrutia, "On optimal coverage of a tree with multiple robots," *European Journal of Operational Research*, vol. 285, no. 3, pp. 844 – 852, 2020.
- [2] D. Kreher and D. Stinson, *Combinatorial Algorithms*. CRC Press, Boca Raton, 1998.
- [3] J. Pallo, "Generating trees with n nodes and m leaves," *International Journal of Computer Mathematics*, vol. 21, no. 2, pp. 133–144, 1987.
- [4] S. Nakano, "Efficient generation of plane trees," *Information Processing Letters*, vol. 84, no. 3, pp. 167 – 172, 2002.
- [5] K. Yamanaka, Y. Otachi, and S.-i. Nakano, "Efficient enumeration of ordered trees with k leaves (extended abstract)," in *WALCOM: Algorithms and Computation*, S. Das and R. Uehara, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 141–150.
- [6] D. Avis and K. Fukuda, "Reverse search for enumeration," *Discrete Applied Mathematics*, vol. 65, no. 1, pp. 21 – 46, 1996, first International Colloquium on Graphs and Optimization.
- [7] D. Knuth, "Generating all trees, history of combinatorial generation," *The Art of Computer Programming, fascicle 4*, vol. 4, 2006.

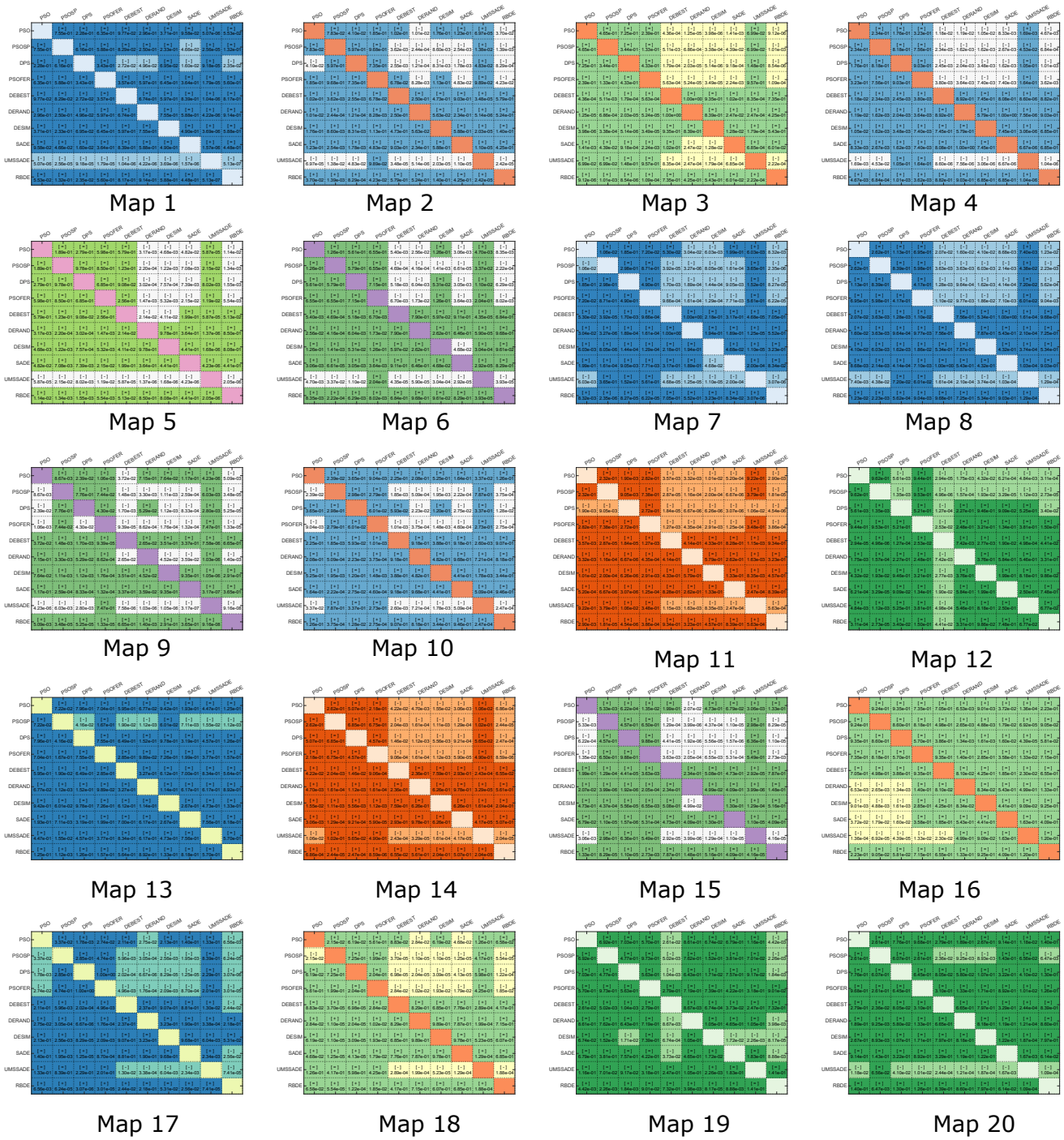


Fig. 12: Statistical significance as measured by the Wilcoxon rank sum test in all maps.

- [8] G. Li and F. Ruskey, “The advantages of forward thinking in generating rooted and free trees,” in *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '99. USA: Society for Industrial and Applied Mathematics, 1999, p. 939–940.
- [9] V. Parque and T. Miyashita, “An efficient scheme for the generation of ordered trees in constant amortized time,” in *15th International Conference on Ubiquitous Information Management and Communication, IMCOM 2021, Seoul, South Korea, 2021*. IEEE, 2021, pp. 1–8.
- [10] J. Sawada, “Generating rooted and free plane trees,” *ACM Trans. Algorithms*, vol. 2, no. 1, p. 1–13, Jan. 2006.
- [11] S.-P. Eu, S. Seo, and H. Shin, “Enumerations of vertices among all rooted ordered trees with levels and degrees,” *Discrete Mathematics*,

- vol. 340, no. 9, pp. 2123 – 2129, 2017.
- [12] M. Atkinson and J. Sack, “Generating binary trees at random,” *Information Processing Letters*, vol. 41, p. 21, 1992.
- [13] J. Yu and Z.-L. Tang, “Generating strictly binary trees at random based on convex polygon triangulations,” *International Journal of Computer Mathematics*, vol. 93, no. 3, pp. 445–452, 2016.
- [14] K. Humphreys, “A history and a survey of lattice path enumeration,” *Journal of Statistical Planning and Inference*, vol. 140, no. 8, pp. 2237–2254, 2010, lattice Path Combinatorics and Applications.
- [15] A. González-Sieira, M. Mucientes, and A. Bugarín, “Motion planning under uncertainty in graduated fidelity lattices,” *Robotics and Autonomous Systems*, vol. 109, pp. 168–182, 2018.

- [16] M. Li, A. Richards, and M. Sooriyabandara, "Asynchronous reliability-aware multi-uav coverage path planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 10023–10029.
- [17] J. Zhao, Z. Jia, Y. Zhou, R. Zhang, Z. Xie, Z. Xu, Y. Li, and D. Zhang, "Path planning based on multi-objective topological map," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, 2021, pp. 1719–1726.
- [18] N. Ma, X. Yu, W.-N. Chen, and J. Zhang, "Fast 3d path planning based on heuristic-aided differential evolution," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 285–286.
- [19] V. Parque and T. Miyashita, "Smooth curve fitting of mobile robot trajectories using differential evolution," *IEEE Access*, vol. 8, pp. 82 855–82 866, 2020.
- [20] R. Oliveira, M. Cirillo, J. M. artensson, and B. Wahlberg, "Combining lattice-based planning and path optimization in autonomous heavy duty vehicle applications," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 2090–2097.
- [21] V. de C. Santos, F. E. B. Otero, C. Johnson, F. S. Osório, and C. F. M. Toledo, "Exploratory path planning for mobile robots in dynamic environments with ant colony optimization," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, ser. GECCO '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 40–48.
- [22] Z. Nie, X. Yang, S. Gao, Y. Zheng, J. Wang, and Z. Wang, "Research on autonomous moving robot path planning based on improved particle swarm optimization," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 2532–2536.
- [23] V. Parque and T. Miyashita, "Towards fast data-driven smooth path planning with fair curves," in *44th IEEE Annual Computers, Software, and Applications Conference, COMPSAC 2020, Madrid, Spain, 2020*. IEEE, 2020, pp. 1115–1116.
- [24] V. Parque, "A study of fairness functionals for smooth path planning in mobile robots," in *17th IEEE International Conference on Automation Science and Engineering, CASE 2021, Lyon, France, 2021*. IEEE, 2021, pp. 1568–1573.
- [25] H. Cuntz, "Bct formalism - trees toolbox," https://treestoolbox.org/manual/BCT_formalism.html, 2021, [Online; accessed 12-December-2021].
- [26] F. H. Eckman, F. E. Theunissen, and J. P. Miller, *NeMoSys: A System for Realistic Single Neuron Modeling*. Boston, MA: Springer US, 1994, pp. 135–145.
- [27] C. Hermann, F. Friedrich, A. Borst, and M. Häusser, "One rule to grow them all: A general theory of neuronal branching and its practical application," *PLOS Computational Biology*, vol. 6, no. 8, pp. 1–14, 08 2010.
- [28] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948 vol.4.
- [29] X. Li, J. Branke, and T. Blackwell, "Particle swarm with speciation and adaptation in a dynamic environment," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 51–58.
- [30] V. Parque, "A differential particle scheme and its application to pid parameter tuning of an inverted pendulum," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1937–1943.
- [31] X. Li, "A multimodal particle swarm optimizer based on fitness euclidean-distance ratio," in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 78–85.
- [32] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, Dec 1997.
- [33] E. Segredo, E. Lalla-Ruiz, and E. Hart, "A novel similarity-based mutant vector generation strategy for differential evolution," in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018, Kyoto, Japan, July 15-19, 2018*, 2018, pp. 881–888.
- [34] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.
- [35] X. Zhou and G. Zhang, "Differential evolution with underestimation-based multimutation strategy," *IEEE Transactions on Cybernetics*, vol. 49, no. 4, pp. 1353–1364, 2019.
- [36] A. M. Sutton, M. Lunacek, and L. D. Whitley, "Differential evolution and non-separability: Using selective pressure to focus search," in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 1428–1435.
- [37] V. Parque, "Towards higher order fairness functionals for smooth path planning," in *GECCO '21: Genetic and Evolutionary Computation Conference, Companion Volume, Lille, France, July 10-14, 2021*, K. Krawiec, Ed. ACM, 2021, pp. 319–320.
- [38] M. Abdelwahab, V. Parque, A. M. R. F. El-Bab, A. A. Abouelsoud, and S. Sugano, "Trajectory tracking of wheeled mobile robots using z-number based fuzzy logic," *IEEE Access*, vol. 8, pp. 18 426–18 441, 2020.
- [39] M. Abdelwahab, V. Parque, A. A. Abouelsoud, and A. M. R. F. El-Bab, "Navigation of omni-directional mobile robot in unstructured environments using fuzzy logic control," in *IEEE/SICE International Symposium on System Integration, SII 2021, Iwaki, Japan, January 11-14, 2021*. IEEE, 2021, pp. 684–689.