An efficient tangent graph based path planning for mutiple topologically distinctive paths

Zhuo Yao, Wei Wang∗

***Abstract*—Conventional local planners frequently become trapped in a locally optimal trajectory, primarily due to their inability to traverse obstacles. Having a larger number of topologically distinctive paths increases the likelihood of finding the optimal trajectory. It is crucial to generate a substantial number of topologically distinctive paths in real-time. Accordingly, we propose an efficient path planning approach based on tangent graphs to yield multiple topologically distinctive paths. Diverging from existing algorithms, our method eliminates the necessity of distinguishing whether two paths belong to the same topology; instead, it generates multiple topologically distinctive paths based on the locally shortest property of tangents. Additionally, we introduce a priority constraint for the queue during graph search, thereby averting the exponential expansion of queue size. To illustrate the advantages of our method, we conducted a comparative analysis with various typical algorithms using a widely recognized public dataset[1]. The results indicate that, on average, our method generates 400 topologically distinctive paths within 250 milliseconds in average. This outcome underscores a significant enhancement in efficiency when compared to existing methods. To foster further research within the community, we have made the source code of our proposed algorithm publicly accessible[2]. We anticipate that this framework will significantly contribute to the development of more efficient topologically distinctive path planning, along with related trajectory optimization and motion planning endeavors.**

***Index Terms*—distinctive topology path, tangent graph, trajectory optimization, path planning.**

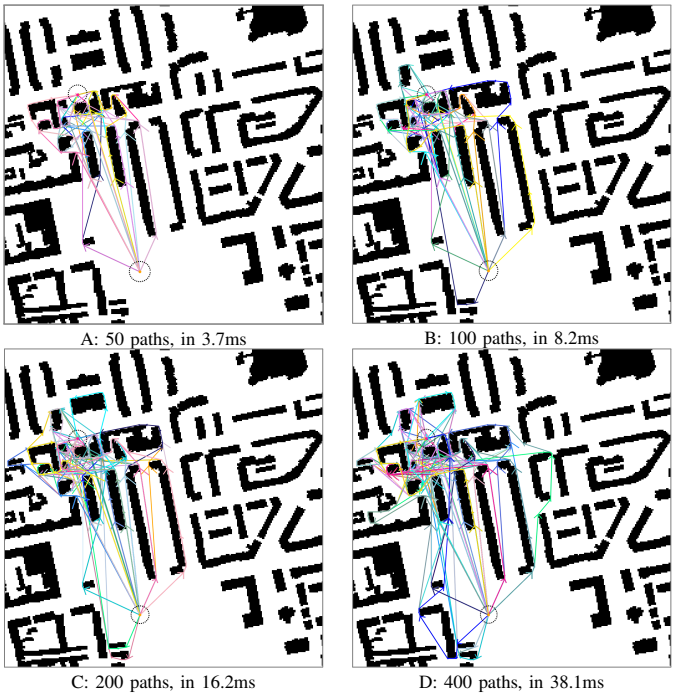


Fig. 1. These figures display multiple paths, with some of them partially overlapping, between the same start and target points, all determined by our method. The map employed is a 256x256 grid map ("Berlin_1_256" from the mentioned public map dataset). The start and target points, highlighted in pink and yellow, respectively, have coordinates (59, 72) and (109, 214), located at the center of circles. These experiments were conducted on a standard laptop running the Ubuntu operating system, equipped with a 3.2GHz CPU and 16GB of memory. Further details can be found in the Results section.

I. Introduction

As trajectory optimization and some motion planning algorithms take an initial path as input, and once a reference path is generated, it cannot be updated to belong to a different topology through gradient-based optimization. And the optimal trajectory under different constraints and requirements may belong to topologically distinctive paths, it is crucial to consider multiple topologically distinctive paths simultaneously.

Rosmann et al. [14] proposed an integrated approach for efficient online trajectory planning of topologically distinctive mobile robot trajectories. Ranganeni et al. [12] integrated user-defined homotopy constraints and footstep planning for humanoid robots, resulting in a significant speedup in planning, particularly in more complex scenarios. Palmieri et al. [11] introduced a Voronoi graph-based topologically distinctive path planning (also known as topology-aware path planning) method to find multiple socially-aware paths, from which the robot can choose the best one based on a social cost function. Kim et al. [4] developed a perception-aware planner that selects the path with the highest perception quality from a set of multiple topologically distinctive paths for MAVs.

However, several difficulties are encountered by existing works. Firstly, some of them require the use of H-signature[2, 3] or similar indicators to determine whether two paths belong to the same topology. This results in an increase in time complexity as the number of paths to be considered grows, as it requires more calculations to determine if a new path and a previous path share the same topology. This results in an increase in time complexity as the number of paths to be considered grows, requiring more calculations to determine if a new path and a previous path share the same topology.

To address the limitations of H-signature[2, 3] and similar indicators, Voronoi-based approaches[6, 11] have been introduced. The distinctive property of Voronoi graph is that differ-

Corresponding by Wei Wang: wangweilab@buaa.edu.cn. This work was supported by the National Key Research and Development Program of China under grant number 2020YFB1313600.

[1]https://movingai.com/benchmarks/grids.html

[2]https://joeyao-bit.github.io/posts/2023/09/07/

ent sequences of Voronoi nodes inherently define topologically distinctive paths. While some of these methods have shown significant improvements in efficiency compared to algorithms that use H-signature. However, these methods still require a sequential search for topologically distinctive paths, one at a time.

To address the aforementioned issues, we propose a novel topologically distinctive path planning algorithm based on tangent graphs. This approach leverages the property that tangents form locally shortest paths, as discussed in previous works [8, 9, 16]. Consequently, any two locally shortest paths inherently belong to different topologies. Our algorithm eliminates the need for indicators to determine path topology similarity and avoids the repetition of searches to obtain multiple paths. Instead, it retrieves all relevant paths in a single search, resulting in a significant improvement in efficiency when compared to existing methods. Additionally, we introduce a priority limitation mechanism to prevent the exponential growth of the queue size during graph search. The results demonstrate a remarkable enhancement in efficiency during the path planning process.A brief demonstration of our algorithm is presented in Fig. 1.

The following sections of this article are organized as follows: Section II offers an introduction to relevant studies on distinctive topology path planning; section III provides a detailed description of the theory basis and key processes involved in our method; in Section IV, we delve into the details of the construction of the tangent graph on the specified maps. Additionally, we present a comprehensive comparison between our method and several typical methods in terms of time cost, success rate and path length. This section also includes information about how our method performs as the number of paths increases and the scale of map increases; finally, in Section V, we discuss the results obtained and potential drawbacks of the proposed method.

This letter contributes the following:

1) A topologically distinctive path planning method based on the locally shortest property of tangents.
2) The introduction of expansion reduction mechanism to mitigate the exponential growth of queue size in path planning that utilizes breadth-first search.

II. Related works

Two trajectories τ_1 and τ_2, connecting the same start and end coordinates, are considered homotopic if one can be continuously deformed into the other without intersecting any obstacles. Otherwise, they are considered topologically distinctive. Based on whether they introduce an explicit indicator, existing topologically distinctive path planning methods can be categorized into two types.

The first type introduces an explicit indicator to determine whether two paths belong to the same topology, typically in relation to obstacles.

H-signature, proposed by Subhrajit Bhattacharya for 2-dimensional maps, is an indicator computed using the Cauchy integral theorem and the Residue theorem from complex analysis. It is defined as the integration of an "obstacle marker function" along a path, where the obstacle marker function involves the representative point of each obstacle. Bhattacharya's work, as seen in references [1, 5, 3], extends H-signature to 2D maps and later to 3D maps [2]. Two paths share the same H-signature if and only if they belong to the same topology; otherwise, they do not. Combining the H-signature constraint with standard graph search algorithms like A*, these methods find multiple topologically distinctive paths by repeating graph searches multiple times.

However, H-signature's efficiency is affected by several factors: 1, the more obstacles there are, the more calculations are needed to compute the H-signature of a path; 2, as the number of paths increases, the average time required to compute a single path increases linearly, as each path needs to be compared with the H-signature of existing paths, and the number of existing paths grows. In contrast, our method does not need an explicit indicator to distinguish whether two paths belong to the same topology. As a result the average time required to compute a single path is increase very slowly as total number of path increases.

Markus Kuderer [6] introduced the concept of the winding angle of a trajectory, defined as the sum of infinitesimal angle differences to the representative points of obstacles along the trajectory. Similar to H-signature, paths belonging to the same topology have the same winding angle, while those with distinctive topologies do not.

To address the issue of multiple paths to the goal within the same homotopy class when using H-signature with A*, Kuderer and colleagues incorporated Voronoi graphs to ensure that each path corresponds to a unique homotopy class. Their results, as presented in [6], showed a significant improvement in efficiency compared to using H-signature combined with A*. In comparing to the Voronoi graph, our method generate locally shortest paths, which are shorter than the paths produced by Voronoi graphs in terms of path length. Our method utilizes tangent graph[8, 10] to generates locally shortest paths.

TARRT* (Topology-Aware RRT*), as proposed by Yi et al. in [18], introduces a method for determining the homotopic equivalence of two arbitrary paths based on properties of strings recognized by a Deterministic Finite Automaton (DFA). These strings also involve the representative point of each obstacle, with the condition that no point is allowed to lie on any line connecting any two other representative points.

Specifically, non-obstacle regions are divided into a set of subregions by lines that intersect representative points, and the connections between these subregions are referred to as reference frames. TARRT* assigns a unique label to each reference frame and appends the label of the reference frames to the string whenever the path crosses them. This string-based approach appears to be more efficient than H-signature because its calculation doesn't involve all obstacle representative points. However, TARRT* shares a limitation with normal RRT (Rapidly Exploring Random Trees) in that they are both probabilistically complete methods for finding existing solutions. For instance, they may fail when the path must pass through a narrow, long opening in an obstacle.

In constrast, our method use a tangent graph to determine all topologically distinctive path, which is complete in de-

termining all possible solutions rather than probabilistic in determining all solutions.

The second type of methods does not rely on an explicit topology indicator.

As previously noted, path segments in the Voronoi graph are inherently unique between two obstacles, ensuring that different paths found in the Voronoi graph guarantee distinctive topology. Consequently, the Voronoi graph is widely employed in distinctive topological path planning.

Luigi Palmieri [11] introduced the Randomized Homotopy Classes Finder (RHCF), a fast randomized method that identifies a set of K paths lying in distinct homotopy classes using the Voronoi graph. The approach involves repeatedly searching for random paths in the Voronoi graph and saving them if they differ from all previous paths. The process stops when K different paths have been found. In their results, RHCF outperformed Yen's K shortest path finding method [17] in terms of speed.

As well as [6], Voronoi graph generate path that far from locally shortest paths, while our method use tangent graph to generate locally shortest path. Additionally, RHCF's mean time cost of search path increase as total number of path increases (more comparison between paths), while our method's average time required to compute a single path is increases very slowly as total number of path increases.

In contrast to mobile robots, which require determining the optimal joint path in the task space, J.J. Rice [13] proposed a bifurcation branch algorithm to characterize the solution space with the bifurcation branch roadmap. This approach generates initial joint paths that identify all homotopy classes and locally optimal paths in all relevant homotopy classes with the lowest cost, which is very likely the globally optimal path. While this method is more capable than traditional approaches in finding the global optimal path, it faces challenges as the number of homotopy classes grows exponentially with the complexity of the solution space. This presents significant difficulties when extending it to topologically distinctive path planning for mobile robots.

In constrast, our method can adapt to complex grid maps, as the exponentially growth of number of homotopy classes is limited by the limitation of queue size during Breadth First Search.

III. Methodology

A. Theory basis

In this section, we aim to establish the relationship between topologically distinct paths and locally shortest paths. We begin by providing clear definitions for locally shortest paths and topologically distinctive paths. Subsequently, we prove the relationship between these two concepts. It is essential to note that all paths discussed herein share the same start and target points.

Definition 1: A path p is considered a locally shortest path if, within any small region along its trajectory, it is impossible to find a shorter collision-free path than p [7].

Definition 2: Two trajectories (or paths), τ_1 and τ_2, are said to belong to the same homotopy class if one can be smoothly deformed into the other without intersecting obstacles; otherwise, they belong to different homotopy classes [1].

Theorem 1: Any two locally shortest paths that share the same start and target cannot belong to the same homotopy class.

Proof 1: Assuming two locally shortest paths (p_1 and p_2) are in the same homotopy class implies that one of them (assume p_1) can be transformed consistently into the other (p_2) without intersecting obstacles. However, during this transformation from p_1 to p_2, the length of the path increases monotonically. This contradicts the definition of a locally shortest path, which is the shortest path within its local neighborhood. Therefore, any nearby path to p_2 should be longer than p_2.

Building upon this theoretical foundation, if multiple locally shortest paths exist, we consequently obtain multiple topologically distinctive paths.

Theorem 2: Locally shortest paths can be formed by tangents connecting the start and target points.

Proof 2: This has been established in previous work [7, 10, 16].

Therefore, utilizing the tangent graph allows for the generation of multiple topologically distinctive paths. Importantly, the time cost of tangent graph detection remains insensitive to the scale increase of maps, as the tangent graph is solely determined by the geometric elements of the maps. These advantages will be demonstrated and discussed in Section IV.

B. Algorithm

Building upon the theoretical foundation, we can extract multiple topologically distinctive paths from the tangent graph. In this section, our focus shifts to the efficient utilization of this graph to obtain multiple topologically distinctive paths. Traditionally, two well-known methods for determining paths between two nodes in a graph are Breadth-First Search (BFS) and Best-First Search. Given that existing methods (refer to Section II for more details) often require multiple searches to obtain multiple paths, our objective is to find multiple paths in a single graph search. Consequently, we choose to implement our method based on Breadth-First Search. However, due to the specialized nature of searching for topologically distinctive paths, certain modifications are necessary.

1) Loop avoidance: As a fundamental principle in topologically distinctive paths [3], path planning with distinctive topology strictly avoids the presence of loops. Loops in this context can take two forms: either a waypoint is revisited within a single path, or the path intersects with itself. It is crucial to consider these scenarios when attempting to find multiple paths. Notably, as the number of waypoints in a path increases, the time required for loop detection also grows. Experimental results demonstrate that the ratio of loop avoidance in the overall time cost increases proportionally with the required number of paths.

2) Path search: Our method comprises two key components: tangent graph detection and path search. Tangent graph detection from grid maps is a well-explored area, and there exist established methods for this purpose [7, 16, 10]. Consequently, our primary focus in this manuscript lies in the efficient search for paths.

Algorithm 1: Topology-aware Breadth First Search

Input: $g_s, g_t, G(V, E), K$
Output: P

$P = \{g_s\}, P' = \emptyset, P_f = \emptyset$;
add g_s and g_t to G;
while *find no* K *paths and* P *is not empty* **do**
  $P' = \emptyset$;
  for *each unfinished path* p *in* P **do**
    for *each nodes* v *of* G *that connect to last nodes of* p **do**
      p' = add v to p;
      if p' *have no loops and* p' *meet taut condition* **then**
        if p' *reach* g_t **then**
          add p' to P';
        else
          add p' to P_f;
  $P = P'$;
return P_f;

TABLE I
TABLE OF NOTATIONS

Name	Meanning	Name	Meanning
g_s	start point	g_t	target point
P, P'	set of unfinished path	P_f	set of finished path
$G(V, E)$	tangent graph	v	a node in tangent graph
p, p'	path	P_B	set of unifinished path (backup queue)

To leverage the efficiency of line-of-sight scans [10], we adopt the ENL-SVG method [10] to generate the tangent graph. To avoid redundant tangent graph computations for the same map, we store the tangent graph in files. Ensuring that all paths are locally shortest, we incorporate the taut condition [10] during the path search.

We have modified the classical Breadth-First Search (BFS) algorithm to search for multiple topologically distinctive paths, as outlined in Algorithm 1. The notation used in the pseudocode is detailed in Table I.

3) Expansion reduction: One drawback of Algorithm 1 is the exponential growth in the size of unfinished paths as the number of expansions increases. To address this issue, we propose an expansion reduction technique to mitigate the exponential growth during iterations of BFS. This technique involves splitting the queue into two: a priority queue and a backup queue.

Assuming the required number of paths is denoted as K, in each iteration of BFS, we select the top K paths with the minimal heuristic value (calculated in the same manner as A*) and store them in the priority queue as candidates for expansion. The remaining unfinished paths are cached in a backup queue, sorted by their heuristic values.

If there are fewer than K unfinished paths in the priority queue, we extract unfinished paths from the backup queue and insert them into the priority queue until there are K elements in the priority queue or the backup queue is empty, as shown in Algorithm 2. The relevant code related to the priority queue has been highlighted.

Algorithm 2: Topology-aware Breadth First Search with expansion reduction

Input: $g_s, g_t, G(V, E), K$
Output: P

$P = \{g_s\}, P' = \emptyset, P_f = \emptyset, P_B = \emptyset$;
add g_s and g_t to G;
while P_f *have no* K *paths and* P *is not empty* **do**
  $P' = \emptyset$;
  for *each unfinished path* p *in* P **do**
    for *each nodes* v *of* G *that connect to last nodes of* p **do**
      p' = add v to p;
      if p' *have no loops and* p' *meet taut condition* **then**
        if p' *reach* g_t **then**
          add p' to P';
        else
          add p' to P_f;
  $P = \emptyset$;
  sort P' in increasing heurstic value;
  move top K path of P' to P and move remaining path in P' to P_B;
  sort P_B in increasing heurstic value;
  if P *have no* K *paths* **then**
    while P_B *is not empty and* P *have no* K *path* **do**
      move path with minimum heurstic value in P_B to P;
return P_f;

Essentially, this approach changes the order of expansion of unfinished paths, prioritizing paths that are closer to the target (i.e., have lower heuristic values). As a result, the path search concludes sooner compared to the scenario without expansion reduction. Importantly, the expansion reduction does not compromise completeness. Algorithm 2 outlines BFS with expansion reduction, specifying its differences compared to Algorithm 1.

An illustrative example highlighting the benefits of introducing expansion limitation is presented in Fig. 2. The map used for this comparison is Berlin_1_256.map from the public grid map dataset mentioned earlier, with the start and target points set at (59, 72) and (109, 214), respectively.

IV. RESULTS

In this section, we delve into a detailed examination of our algorithm's performance, comparing it with other distinctive topology path planning methods. The first subsection provides insights into the construction of the tangent graph. To ensure a fair comparison, we utilize the implementations provided by the respective authors. Our method is compared with HA*,

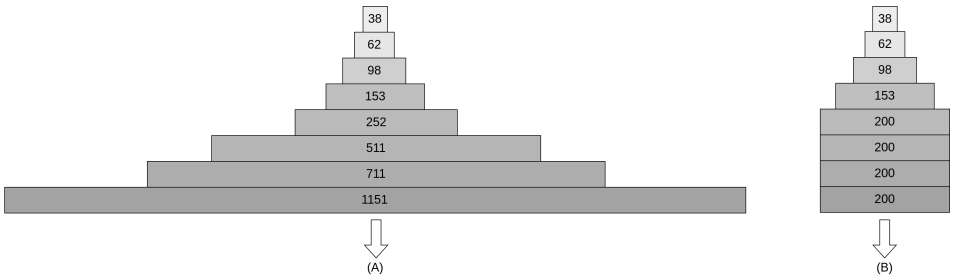


Fig. 2. These figures depict the changes in queue size during BFS expansion before (Fig A) and after (Fig B) implementing expansion reduction in the same scenario (Fig. 1), where the goal is to find 200 paths. In both Figures A and B, it's evident that the queue size grows exponentially before the introduction of priority limitation, but stabilizes once it reaches 200 after implementing this enhancement. Consequently, obtaining 200 paths takes 48.5ms before the introduction of priority limitation, while it only takes 16.8ms afterward.

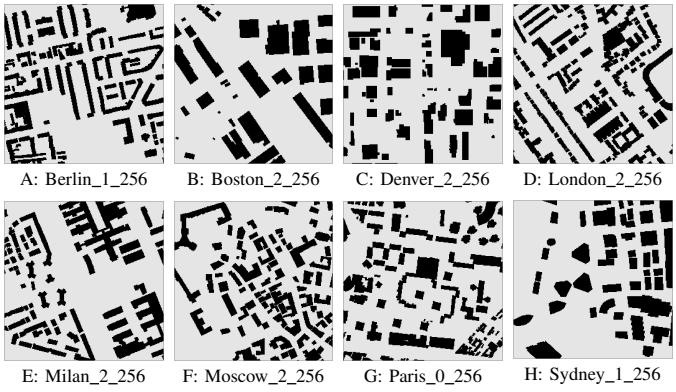
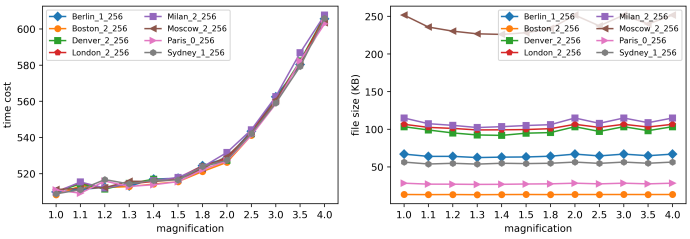


Fig. 3. These figures showcase eight grid maps from [15] utilized in the comparison with other algorithms. The size of each map is 256×256.



A: time cost comparison B: time cost comparison

Fig. 4. These two figures present comprehensive data pertaining to the tangent graph detection process of our method. The data includes information on time cost and file size to save the graph, encompassing various magnifications of the map. Figure 1 details the time cost across different magnifications, while Figure 2 provides insights into the corresponding file sizes.

HTheta*, and RHCF, focusing on the time cost as the required number of paths and the scale of the map increases. Path length comparisons for each method are also conducted. HA* and HTheta* are H-signature-based algorithms [3], while RHCF is a Voronoi graph-based algorithm [4].

The experiments are conducted using a well-known grid map dataset [15] [5]. City maps from the dataset are selected for their high obstacle density, allowing the generation of hundreds of topologically distinctive paths, providing ample space for each method to demonstrate its capability. For each map, 100 start and target combinations are randomly sampled as inputs for path planning. To study how method performance changes with an increasing map scale, raw maps are magnified to create larger maps. The magnification ranges from 1.0 to 4.0 (including 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.8, 2.0, 2.5, 3.0, 3.5, 4.0 specifically), resulting in map scales from 256×256 to 1024×1024. Experiments were conducted on a laptop running Ubuntu 20.04, equipped with a Ryzen 7 5800h (3.2GHz) CPU and 16GB of memory.

To showcase the methods' capability to find a substantial number of topologically distinctive paths, we set all methods to find paths ranging from 10 to 400 (including 10, 20, 30, 40, 50, 80, 120, 160, 200, 250, 300, 350, 400 specifically). For HA* and HTheta*, their success rate drops to less than 10% when attempting to find 200 topologically distinctive paths, so we set the maximum number of required paths for them to 200 to reduce the overall time cost of the experiment.

A. Construction of tangent graph

In this section, we elaborate on the construction of the tangent graph for the aforementioned eight maps across various magnifications. The associated data, including time cost and file size, is summarized in Fig 4.

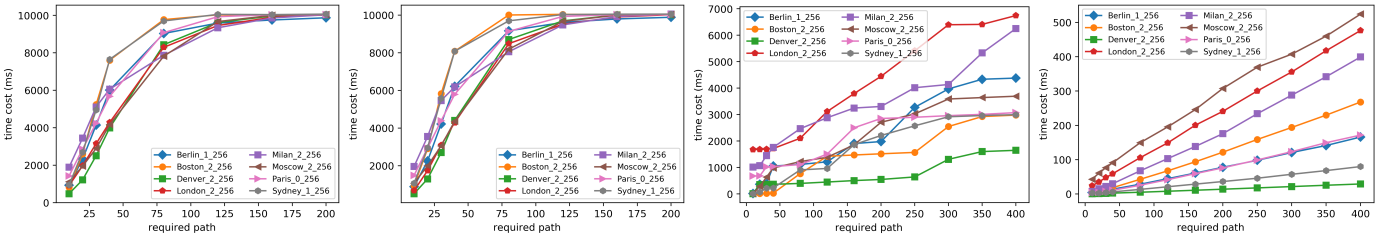
The construction of the tangent graph for nearly all maps is completed in approximately 0.5 seconds, indicating nearly real-time updates. Notably, the time cost exhibits a gradual increase as the scale magnifies. Specifically, when the map's scale increases fourfold, the time cost only experiences a 20% increment.

The file size required to store the tangent graph for these maps ranges from 10KB to 300KB, demonstrating a manageable size for ordinary computing platforms. Importantly, the size of the tangent graph remains stable as the map scale increases. This observation suggests that the tangent graph exhibits insensitivity to variations in map resolution, making it well-suited for real-world maps with high resolutions.

B. Comparison with other methods

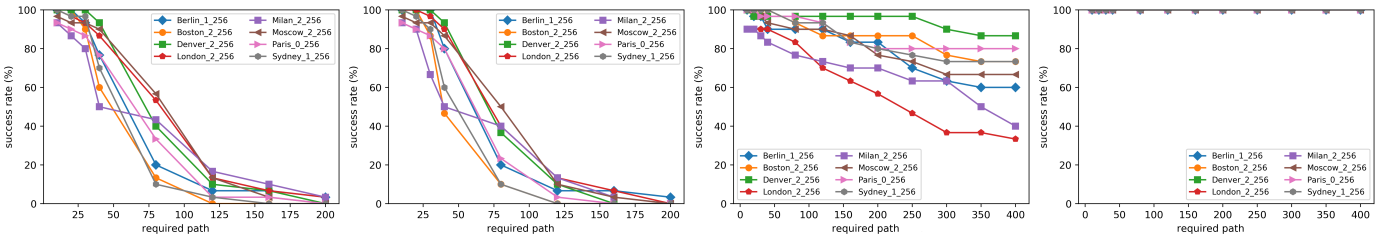
In this section, our focus centers on the comparative analysis of our method with other prominent approaches, namely HA*, HTheta*, and RHCF. In practical applications, topologically distinctive path planning is often constrained by an upper limit

[3]https://github.com/subh83/DOSL
[4]https://github.com/srl-freiburg/srl_rhcf_planner
[5]https://movingai.com/benchmarks/grids.html



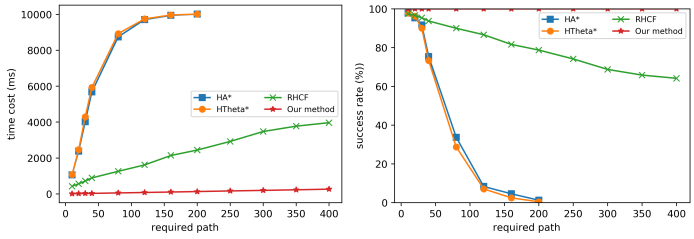
A: HA* B: HTheta* C: RHCF D: Our method

Fig. 5. These figures illustrate the mean time cost for finding the required number of paths for each map using all four methods. The magnification for all maps is set to 1.0.



A: HA* B: HTheta* C: RHCF D: Our method

Fig. 6. These figures illustrate the mean success rate for finding the required number of paths for each map using all four methods. The magnification for all maps is set to 1.0.



A: time cost comparison B: success rate comparison

Fig. 7. These figures show the comparison of the success rates and time costs of the four methods across all eight maps, with a consistent magnification set to 1.0.

on time costs to avoid exceeding expected time constraints. When a method fails to find all solutions within the specified time bound, it returns the solutions it has discovered up to that point. Therefore, in this section, we impose a time upper bound of 10 seconds and employ the success rate – the ability to find all solutions within the time constraint – as an indicative measure for evaluating the performance of the methods. Essentially, the success rate is determined by the time cost incurred.

Furthermore, we conduct a comparative assessment of all methods with respect to path length – a crucial metric in path planning. Recognizing that map scale significantly influences the performance of path planning algorithms, we conduct tests on the four aforementioned methods using maps with identical content but scales ranging from 256×256 to 1024×1024.

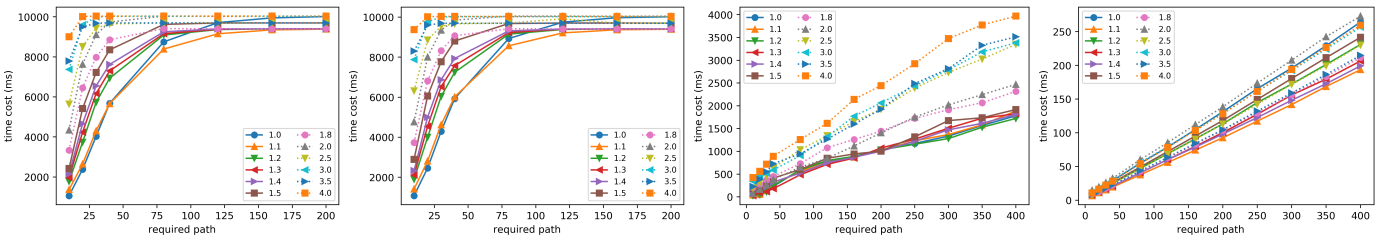
1) Time cost and success rate: Initially, we subjected all four methods to testing under the specified eight maps, utilizing them to search varying numbers of topologically distinctive paths (ranging from 10 to 400). The mean time cost and success rate were recorded for each map, as depicted in Fig. 5 and Fig. 6 respectively. The amalgamated results, showcasing the interplay between time cost and success rate, are illustrated in Fig. 7.

As illustrated in Fig. 5 and Fig. 6, our method achieves a mean time cost of approximately 500ms to find 400 paths, boasting a 100% success rate. In contrast, RHCF takes around 7s to find 400 paths on average, with its success rate varying from 80% in the best case to 40% in the worst case. For HA* and HTheta*, their mean time cost approaches the time upper bound (10s) when attempting to find more than 80 paths. Moreover, their success rate declines as the mean time cost increases, dropping to 50% after attempting to find more than 40 paths. Notably, our method exhibits a slower increase in time cost compared to other methods, as it circumvents multiple graph searches for multiple paths, obtaining all required paths in a single search. With an average time cost of only 500ms in the worst case, our method consistently achieves a 100% success rate in finding the required number of paths.

This section underscores our method's substantial advantage over other methods in terms of time cost and success rate.

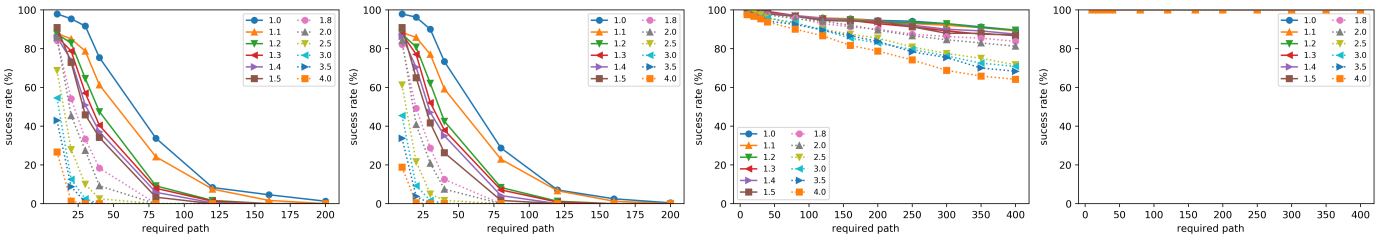
2) Path length: Path length, a critical attribute in path planning, is the focus of comparison among all four methods in this section. However, it's worth noting that methods like HA* and HTheta*, which incur longer computation times for extended paths, may prioritize finding shorter paths and consequently may not reach the required number of paths. To ensure fairness in our comparison, we exclusively examine path lengths when finding 10 topologically distinctive paths, a scenario where all methods achieve a success rate exceeding 90%, as depicted in Fig. 10.

Notably, RHCF exhibits the longest paths, given its reliance on the Voronoi Graph. Both HTheta* and HA* yield shorter paths than RHCF, as they are rooted in A*-based shortest path finding. Interestingly, HTheta* and our method achieve shorter paths than HA*, leveraging tangent graph-based path planning and the any-angle shortest path planning characteristic of HTheta*, whereas A* relies on four-neighbor or eight-



A: HA* B: HTheta* C: RHCF D: Our method

Fig. 8. These figures illustrate that the mean time cost to find the required number of paths increases with the magnification of the maps, covering all eight maps.



A: HA* B: HTheta* C: RHCF D: Our method

Fig. 9. These figures illustrate that the mean success rate to find the required number of paths increases with the magnification of the maps, covering all eight maps.

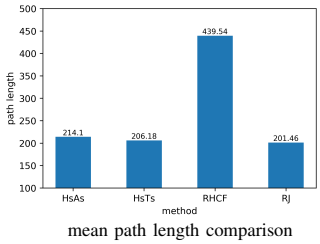


Fig. 10. These figures present a comparison of path lengths among the four methods when the required number of paths is set to 10. The results from all maps are combined, with a consistent magnification of 1.0 for all maps.

neighbor expansion.

3) Map scale and time cost: In this section, our focus centers on exploring the relationship between mean time cost and map scale. We conducted tests for all four methods across eight maps with varying magnifications, recording the mean time cost and success rate for different required numbers of paths, as depicted in Fig. 8 and Fig. 9.

As evident in these figures, HA* and HTheta*'s time cost increases significantly, and their success rate decreases as the map magnification scales from 1.0 to 4.0. This decrease is more pronounced as the larger map size requires more node expansions and H-signature calculations.

RHCF exhibits a similar trend, although in a milder manner. The size of the Voronoi graph remains relatively unchanged, but determining connections from start/target to the graph takes more time with larger maps.

In contrast, our method demonstrates a gradual increase in time cost. The main factors influencing its cost are the number of required paths and the size of the tangent graph, which remains almost constant with increasing map scale. The slight fluctuations in our method's cost are attributed to the minor volatility in the CPU's computational performance.

In conclusion, our method demonstrates stability with increasing map scale. It incurs a gradual increase in time cost while maintaining a 100% success rate, contrasting sharply with the notable decline in performance observed in other methods. The resilience of our method as map scale increases can be ascribed to the consistent behavior of the tangent graph, which remains stable in both size and time cost of construction, responding adeptly to variations in scale.

V. Discussion and Conclusion

In this article, we introduce a tangent graph-based topologically distinctive path planning algorithm. It leverages the property that tangents form locally shortest paths, and any two locally shortest paths belong to different topologies. Compared to existing algorithms, our approach requires no indicator to determine whether two paths belong to the same topology. Furthermore, it eliminates the need to repeat the search for multiple paths, as all distinctive paths can be found in a single search.

To address the challenge of the exponential growth in queue size during breadth-first search, we propose an expansion reduction technique. This approach significantly reduces the time cost of searching for multiple paths while preserving computational complexity.

Our method consists of two main steps: the construction of the tangent graph and the search for multiple paths. We employ the ENL-SVG technique, as introduced by [10], to construct the tangent graph, leveraging the advantages offered by line-of-sight scans. Additionally, to avoid repeating these calculations every time the map is loaded, we save the results to a binary file, which is then loaded during the framework loading process, saving valuable time.

During the search for multiple paths, we introduce loop avoidance to prevent the repetition of waypoints in the path

and avoid intersections with itself. Additionally, we apply the taut condition of [10] to ensure every path is locally shortest. It's important to note that as the number of waypoints in the path increases, the time cost of loop detection also increases. Experimental results show that the ratio of no-loop constraint checks in the total cost increases as the required number of paths increases. This occurs because the requirement for more paths often results in paths with a larger number of waypoints.

In comparison with other methods, our approach demonstrates significant efficiency when compared to RHCF, HA*, and HTheta*. Specifically, our method takes approximately 250ms to determine 400 paths from the mentioned public map dataset, while RHCF takes more than 1 second, and HA* and HTheta* take more than 10 seconds. And our method's performance is stable as the scale of map increases. However, it's important to note that our method has a limitation in that it cannot search for multiple paths with the same topology, unlike methods that use indicators like H-signature, which are capable of searching for multiple paths with the same topology.

In the future, we plan to research how to apply our method to 3D maps and implement our method using multiple threads, as currently, it operates in a single thread. Since there is no dependence between different unfinish paths during the search process, there will be no loss of complexity when utilizing multiple threads. Additionally, we intend to apply our method to trajectory optimizations. Given that our method can provide hundreds of topologically distinctive paths in real time, it is crucial to study how to speed up trajectory optimizations to enable them to efficiently handle hundreds of paths as input.

ACKNOWLEDGMENTS

The first author thanks Lu Zhu for her encouragement and support during the consummation of this work.

REFERENCES

[1] Subhrajit Bhattacharya. "Search-based path planning with homotopy class constraints". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 24. 1. 2010, pp. 1230–1237.

[2] Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. "Search-based path planning with homotopy class constraints in 3D". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 26. 1. 2012, pp. 2097–2099.

[3] Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. "Topological constraints in search-based robot path planning". In: *Autonomous Robots* 33 (2012), pp. 273–290.

[4] Dabin Kim et al. "Topology-guided path planning for reliable visual navigation of MAVs". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 3117–3124.

[5] Soonkyum Kim et al. "Optimal trajectory generation under homology class constraints". In: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE. 2012, pp. 3157–3164.

[6] Markus Kuderer et al. "Online generation of homotopically distinct navigation paths". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 6462–6467.

[7] Y.-H. Liu and S. Arimoto. "Proposal of tangent graph and extended tangent graph for path planning of mobile robots". In: *Proceedings. 1991 IEEE International Conference on Robotics and Automation*. 1991, 312–317 vol.1. DOI: 10.1109/ROBOT.1991.131594.

[8] Y-H Liu and Suguru Arimoto. "Proposal of tangent graph and extended tangent graph for path planning of mobile robots". In: (1991), pp. 312–317.

[9] Yun-Hui Liu and Suguru Arimoto. "Path planning using a tangent graph for mobile robots among polygonal and curved obstacles: Communication". In: *The International Journal of Robotics Research* 11.4 (1992), pp. 376–382.

[10] Shunhao Oh and Hon Wai Leong. "Edge n-level sparse visibility graphs: Fast optimal any-angle pathfinding using hierarchical taut paths". In: *Tenth Annual Symposium on Combinatorial Search*. 2017.

[11] Luigi Palmieri, Andrey Rudenko, and Kai O Arras. "A fast randomized method to find homotopy classes for socially-aware navigation". In: *arXiv preprint arXiv:1510.08233* (2015).

[12] Vinitha Ranganeni, Oren Salzman, and Maxim Likhachev. "Effective footstep planning for humanoids using homotopy-class guidance". In: *Proceedings of the International Conference on Automated Planning and Scheduling*. Vol. 28. 2018, pp. 500–508.

[13] Jacob J Rice and Joseph M Schimmels. "Multi-homotopy class optimal path planning for manipulation with one degree of redundancy". In: *Mechanism and Machine Theory* 149 (2020), p. 103834.

[14] Christoph Rösmann, Frank Hoffmann, and Torsten Bertram. "Integrated online trajectory planning and optimization in distinctive topologies". In: *Robotics and Autonomous Systems* 88 (2017), pp. 142–153.

[15] N. Sturtevant. "Benchmarks for Grid-Based Pathfinding". In: *Transactions on Computational Intelligence and AI in Games* 4.2 (2012), pp. 144 –148. URL: http://web.cs.du.edu/~sturtevant/papers/benchmarks.pdf.

[16] Zhuo Yao et al. "ReinforcedRimJump: Tangent-based shortest-path planning for two-dimensional maps". In: *IEEE Transactions on Industrial Informatics* (2019).

[17] Jin Y Yen. "Finding the k shortest loopless paths in a network". In: *management Science* 17.11 (1971), pp. 712–716.

[18] Daqing Yi et al. "Topology-aware RRT* for parallel optimal sampling in topologies". In: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2017, pp. 513–518.