

Homotopy-Aware Multi-Agent Path Planning on Plane

KAZUMI KASAURA, OMRON SINIC X Corporation, Japan

We propose an efficient framework using Dynnikov coordinates for homotopy-aware multi-agent path planning in planar domains that may contain obstacles. We developed a method for generating multiple homotopically distinct solutions for the multi-agent path planning problem in planar domains by combining our framework with revised prioritized planning and proved its completeness under specific assumptions. Experimentally, we demonstrated that our method is significantly faster than a method without Dynnikov coordinates. We also confirmed experimentally that homotopy-aware planning contributes to avoiding locally optimal solutions when searching for low-cost trajectories for a swarm of agents in a continuous environment.

JAIR Track: Multi-Agent Path Finding

JAIR Associate Editor: Daniel Harabor

JAIR Reference Format:

Kazumi Kasaura. 2026. Homotopy-Aware Multi-Agent Path Planning on Plane. *Journal of Artificial Intelligence Research* 85, Article 15 (February 2026), 31 pages. DOI: [10.1613/jair.1.19243](https://doi.org/10.1613/jair.1.19243)

1 Introduction

Path planning with topological constraints or objectives is an important task, because considering topological characteristics of paths is crucial for various aspects of robot planning and navigation. One notable example is when seeking globally optimal trajectories. To find a globally optimal trajectory under a complex objective, (such as considering agent kinematics), a conventional strategy involves planning an initial path on a simple graph (e.g., a grid) and optimizing it locally under the objective function (Rösmann et al. 2017b). However, this can lead to local optima, and it is difficult to know beforehand which path will converge to a globally optimal solution. To avoid this, multiple paths should be optimized and compared. On the other hand, it is redundant to optimize paths that converge to the same solution. Here, the key is topological characteristics of the paths since optimization does not alter them. It has been proposed to generate several topologically distinct paths as initial solutions (Kuderer et al. 2014; Rösmann et al. 2017a). Figure 1 shows an example of initial paths with the same start and goal. Because p_1 goes through the area above the obstacle and p_2 and p_3 go through the area below, p_1 and p_2 (or p_3) have different topological features, while p_2 and p_3 have the same ones. These facts will be more formally discussed later in § 3.1. Since p_1 and p_2 (or p_3) must converge to different trajectories after optimization and, the better option depends on the objective function. Therefore, we want to generate both. Conversely, since p_2 and p_3 are topologically identical and likely converge to the same trajectory, we do not generate both. To do this, planning should be performed with the calculation of the topological characteristics of paths, which we call *topology-aware* path planning. Topology-awareness in planning also finds relevance in tasks such as cable manipulation (Bhattacharya, Kim, et al. 2015; Kim et al. 2014), human-robot interaction (Govindarajan et al. 2016; Yi et al. 2016), and high-level planning with dynamic obstacles (Cao et al. 2019).

Author's Contact Information: Kazumi Kasaura, kazumi.kasaura@sinicx.com, ORCID: [0000-0002-3219-9961](https://orcid.org/0000-0002-3219-9961), OMRON SINIC X Corporation, Tokyo, Japan.



This work is licensed under a [Creative Commons Attribution International 4.0 License](https://creativecommons.org/licenses/by/4.0/).

© 2026 Copyright held by the owner/author(s).

DOI: [10.1613/jair.1.19243](https://doi.org/10.1613/jair.1.19243)

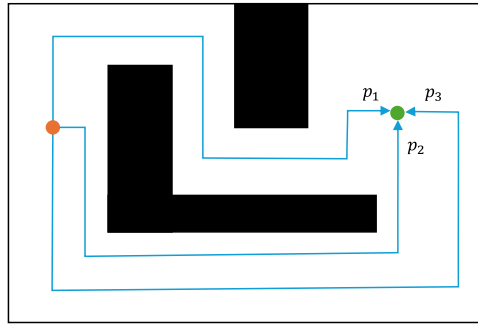


Fig. 1. Example of initial coarse paths connecting the same start and goal. After optimizing, p_1 and p_2 must converge to different smoothed trajectories, while p_2 and p_3 are likely to converge to the same one.

Homotopy is a straightforward topological feature of paths, but difficult to calculate due to its non-abelian nature, as will be discussed later. [Bhattacharya and Ghrist \(2018\)](#) proposed an approach on the basis of a concept called the *homotopy-augmented graph* for general *homotopy-aware* path planning (path planning with the calculation of homotopy classes of paths). Homotopy-aware path planning using a roadmap is reduced to pathfinding on the homotopy-augmented graph constructed from the roadmap.¹ To search on a homotopy-augmented graph, we have to manage elements of the *fundamental group* of the searched space. The fundamental group is not generally abelian and can be computationally difficult to deal with. Elements of the group are represented by strings of generators, called *words*. However, different words can represent the same element of the group, and determining the identity of such representations, termed the *word problem*, is not generally solvable ([Novikov 1955](#)). Consequently, homotopy-aware path planning remains a generally difficult task.

On the other hand, multi-agent path planning, which plans paths for multiple agents so that they do not collide with each other, has a variety of applications ([Dresner and Stone 2008](#); [Li et al. 2020](#); [Silver 2005](#)). In multi-agent scenarios on a plane, topological features are nontrivial even in the absence of obstacles since agents are obstacles for other agents. For example, when two agents pass each other, the topological characteristics of the solution vary with the direction of avoidance (counterclockwise or clockwise) as shown in Figure 2. Thus, topological considerations are important for finding the optimal trajectory described in the first paragraph, even in the cases of obstacle-free environments. Moreover, homotopical considerations are important in the multi-agent case for other reasons. Homotopy is used for multi-agent coordination ([C. I. Mavrogiannis and Knepper 2019](#)), especially for avoiding deadlocks ([Čáp, Gregoire, et al. 2016](#)). However, studies on homotopy-aware planning for multi-agent scenarios, even in obstacle-free environments, are limited.

By combining the aforementioned concepts with research for braids, we obtained an efficient framework for homotopy-aware multi-agent path planning in planar domains, which is the first valid solution to this problem to our knowledge. There are three key ideas. First, while the fundamental group for the multi-agent path planning problem on a plane is the *pure braid group*, we label the homotopy classes of solutions by elements of the *braid group*. This is equivalent to expanding the space of paths to solutions of *unlabeled* multi-agent path planning, in which, as long as each goal is reached by only one agent, it is permissible for any agent to proceed to any goal. ([Adler et al. 2015](#))². This simplifies the word construction. Note that this expansion is only virtual for the

¹Since we focus on the approach of using roadmaps for path planning, the problem that we treat in this paper is actually pathfinding on graphs. However, the homotopies we consider are not those in discrete graphs ([Ghrist 1999](#)) but those in the continuous domain. Thus, to avoid confusion, we do not use the term *homotopy-aware pathfinding*.

²This setting is also called *permutation-invariant* ([Kloder and Hutchinson 2006](#); [Yu and LaValle 2013](#)) or *anonymous* ([Stern et al. 2019](#)).

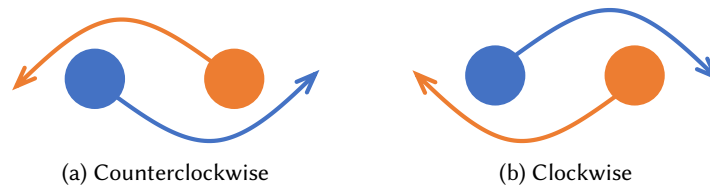


Fig. 2. Two ways for two agents to pass each other. The homotopy class of the path in the configuration space depends on the direction of their passing.

calculation of homotopy classes and the actual pathfinding remains labeled. Since homotopy classes inherently contain information about agent-goal correspondences, this idea does not give rise to any confusion. The second idea is to use *Dynnikov coordinates*, a representation of elements of the braid group as tuples of integers, which are expected to be calculated at a low computational cost (Dehornoy 2008). Thanks to these coordinates, we can efficiently maintain a homotopy-augmented graph by using a data structure such as self-balancing binary search trees (Knuth 1998). The third is that obstacles in the domain are taken into account by considering them as virtual agents.

We consider the problem of generating multiple homotopically distinct solutions for multi-agent path planning in a planar domain, and provide a method for this problem by combining our framework with the *revised prioritized planning* (Čáp, Novák, et al. 2015). We also prove that our method can generate solutions belonging to all homotopy classes under certain assumptions.

We experimentally demonstrate that the runtime of our method increases roughly quadratically with respect to the number of agents on a scale of several hundred agents, while the runtime of another method, which uses the Dehornoy order (Dehornoy 1994) to manage braids instead of Dynnikov coordinates, increases approximately quintically.

In addition, we demonstrated the usefulness of homotopy-awareness for multi-agent trajectory optimization through experimentation. Specifically, we generated several homotopically different (discrete) solutions using our method. We then continuously optimized them for a certain cost function, and chose the best one. We confirmed that this solution was a globally better trajectory than those generated by baseline methods.

The contributions of this paper are summarized as follows:

- We propose the first sound framework for homotopy-aware multi-agent path planning on the plane.
- We provide a method using this framework to generate multiple homotopically distinct solutions for multi-agent path planning in the plane. We have theoretically proved a kind of completeness of our method and experimentally showed its scalability.
- We experimentally showed that solving this problem contributes to multi-agent trajectory optimization.

2 Related Work

We survey previous works for multi-agent path planning, topology-aware path planning, and braids.

2.1 Multi-Agent Path Planning

Multi-agent pathfinding, a field of study focusing on planning for multiple agents in discrete graphs, has been a subject of extensive research, particularly in grid-based environments (Stern et al. 2019). One of the approaches to this problem is prioritized planning (Erdmann and Lozano-Perez 1987; Silver 2005), which is non-optimal, incomplete yet scalable. Čáp, Novák, et al. (2015) proposed Revised Prioritized Planning (RPP) and proved its completeness under certain assumptions. Major optimal approaches for multi-agent pathfinding include

increasing-cost tree search (Sharon, Stern, Goldenberg, et al. 2013), conflict-based search (Sharon, Stern, Felner, et al. 2015), and some reduction-based methods (Barták et al. 2017; Surynek et al. 2016). Surveys of solutions have been conducted by Stern (2019) and Lejeune et al. (2021). While these methods do not consider homotopy, some of them could be modified to do so by combining them with our framework. In this paper, RPP was adopted for the scalability reason. On the other hand, we believe that some methods are not suitable for considering homotopy. See also § 7.

For multi-agent path planning in continuous environments, the typical strategy involves a three-step process: roadmap generation, discrete pathfinding, and continuous smoothing of trajectories. For instance, Hönig et al. (2018) presented such an approach for quadrotor swarm navigation. Since the quality of the smoothed trajectories is difficult to predict beforehand, generating multiple initial solutions, as described in § 1, can be effective. Several roadmap-generation methods tailored for multi-agent scenarios have been proposed (Arias et al. 2021; Henkel and Toussaint 2020; Okumura et al. 2022). A number of multi-agent pathfinding algorithms have been adapted to handle continuous time scenarios (Andreychuk et al. 2022; Surynek 2019; T. T. Walker et al. 2018; Yakovlev and Andreychuk 2017). Although we used a simple grid for simplicity in our experiments, our method can be extended to treat general roadmaps and continuous time. See Remark 2.

2.2 Topology-Aware Path Planning

For the single-agent case on a plane, there are various studies, both theoretical and applied, on homotopy-aware path planning. For planning on a plane with polygonal obstacles, there exist methods using polygon partition (Liu et al. 2023; Park et al. 2015) with analysis of time complexity (Bespamyatnikh 2003; Efrat et al. 2006; Hernandez et al. 2015). For scenarios involving possibly non-polygonal obstacles, several approaches have been explored (Hernandez et al. 2015; Jenkins 1991; Schmitzberger et al. 2002; Yi et al. 2016). Grigoriev and Slissenko (1997, 1998) proposed a method of constructing words by detecting traversing *cuts*, which are also called *rays* (Tovar et al. 2010). Along this idea, the notion of the *homotopy-augmented graph* (*h-augmented graph*), was proposed and applied to the navigation of a mobile robot with a cable (Bhattacharya, Kim, et al. 2015; Kim et al. 2014).

The last approach was generalized by Bhattacharya and Ghrist (2018) to various situations including multi-agent path planning on a plane. However, their algorithm is incomplete. While they attempted to solve the word problem by using Dehn's algorithm (Lyndon et al. 1977), they acknowledged that this algorithm may not always yield accurate results for their presentation.³ To our knowledge, there exists no known complete framework for considering homotopy in the multi-agent case on a plane, even when obstacles are absent.

Homology serves as a coarser classification compared with homotopy, whereby two paths belonging to the same homotopy class are also in the same homology class, but the reverse is not always true⁴. While homology is not well-suited for detailed path analysis like homotopy, it possesses the advantage of being computationally easier due to its abelian nature. Algorithms have been developed for homology-aware path planning in two, three, or higher dimensional Euclidean spaces with obstacles (Bhattacharya 2010; Bhattacharya, Likhachev, et al. 2011, 2012; Bhattacharya, Lipsky, et al. 2013). In the planar scenario, homology can be determined by a tuple of winding numbers (Vernaza et al. 2012). This number count the number of times that an agent travels around an obstacle. In the multi-agent planning in the plane, homology can also be determined by a tuple of all winding numbers between agents or between agents and obstacles (Rolfesen 2010). These facts have been applied to enable homology-aware planning for mobile robot navigation (Kuderer et al. 2014; C. Mavrogiannis and Knepper 2021;

³Actually, their algorithm fails to perform correctly when dealing with scenarios involving more than three agents (see Appendix A for details).

⁴In the context of robotics, the terms "homotopy" and "homology" are sometimes used interchangeably.(Bhattacharya, Likhachev, et al. 2012)

C. I. Mavrogiannis and Knepper 2020). The two solutions in Figure 10 in § 5.3 belongs to the same homology class since they have the winding numbers for all agent pairs, while they are homotopically distinct.

Jaillet and Siméon (2008) introduced the notion of *visibility deformation* (VD), which is stricter than homotopy equivalence: while two paths are homotopic if they are visibility deformable one into the other, the reverse is not always true. Since it is computationally expensive, Zhou et al. (2020) introduced a simpler and even stricter notion, *uniform visibility deformation* (UVD): two paths γ, γ' with the same endpoints belong to the same UVD class if, for any t , the segment between $\gamma(t)$ and $\gamma'(t)$ is contained within the free space. While these notions are useful in three-dimensional environments (Zhou et al. 2020), they will be too strict in two-dimensional environments for our motivation, because path optimization may not be visibility deformation. Moreover, these relations are not equivalence relations in general, because, even if the segment between A and B and the segment between B and C are contained within the free space, the segment between A and C may not.

As mentioned in the introduction, several works (De Groot et al. 2024; Kuderer et al. 2014; Rösmann et al. 2017a; Zhou et al. 2020) focused on the strategy to generate multiple topologically distinct trajectories for avoiding local optima. For navigation with presence of other agents, it is proposed to enumerate topological patterns of agent coordination and to select the best one, by using winding numbers (C. Mavrogiannis and Knepper 2021; C. I. Mavrogiannis and Knepper 2020), which represent homology classes. It was also proposed to use supervised learning in selecting topology class to imitate human behaviors (Martinez-Baselga et al. 2024). Our work will enable the extension of these approaches to multi-agent path planning on the plane, without losing the fineness of homotopical classification.

2.3 Braids and their Applications to Robotics

The braid group was introduced by Artin (1947a,b). Fox and Neuwirth (1962) proved that it is the fundamental group of the unlabeled configuration space. There are several algorithms for solving the word problem of braid groups (Birman et al. 1998; Epstein 1992; Garber et al. 2002; Garside 1969; Hamidi-Tehrani 2000). Dehornoy (1994) introduced a linear order of braids called the *Dehornoy order*, for which comparison algorithms were presented (Dehornoy 1997; Maljutin 2004). Dehornoy et al. (2008) conducted a survey of orders of braids and their comparison algorithms. It was proved that the braid groups are linear (Bigelow 2001; Krammer 2002). Dynnikov coordinates were introduced by Dynnikov (2002).

In robotics, the concept of braid groups on graphs is used for robot planning on graphs (Ghrist 1999; Kurlin 2012). Regarding the planar case, Diaz-Mercado and Egerstedt (2017) developed a framework enabling the control of agents such that their trajectories correspond to specific braids. In their setting, agents move in circular paths on a predefined track, while our approach deals with path planning with arbitrary start and goal positions.

It was proposed to impose homotopy constraints when executing multi-agent plans to prevent deadlock even in the event of delays (Čáp, Gregoire, et al. 2016; Gregoire et al. 2013). Homotopy constraints were also suggested for use in game-theoretic motion planning in urban driving scenarios (Khayyat et al. 2024). Although these studies did not use the braid group, they could be expanded to address more intricate topological relationships via braids.

Braid groups were used for predicting trajectories of other agents in distributed multi-agent navigation (C. I. Mavrogiannis and Knepper 2019). This framework was applied to intersection management (C. Mavrogiannis, DeCastro, et al. 2022). While we focus on a centralized planning for a relatively large number of agents, our efforts to compute braids efficiently could benefit research in this area.

Lin and McCann (2021) used braids to represent states of a knitting machine to find its optimal plans. While their aim is quite different from ours, their proposed method is similar to ours since it searches for optimal transfer plans for states represented by braids. While they used symmetric normal forms (Dehornoy 2008) to manage braids, we do not for the reason of computational efficiency.

3 Preliminary

In § 3.1, we outline some general notions on topology. In § 3.2, we describe the homotopy-augmented graph, which is a specific notion to homotopy-aware path planning. In § 3.4, we describe the braid group, which represents homotopy classes for multiple agents on the plane. To do this, we explain the notion of presentation of a group in § 3.3.

3.1 Homotopy and Fundamental Group

Let X be a topological space. A continuous morphism from the interval $[0, 1]$ to X is called a *path*. Two paths with the same endpoints are said to be *homotopic* if, intuitively, one can be continuously deformed into the other in X . The formal definition is as follows: Two paths γ_0 and γ_1 are homotopic if there exists a continuous map $h : [0, 1] \times [0, 1] \rightarrow X$ such that

$$h(s, 0) = \gamma_0(0) = \gamma_1(0), h(s, 1) = \gamma_0(1) = \gamma_1(1) \quad (1)$$

$$h(0, t) = \gamma_0(t), h(1, t) = \gamma_1(t), \quad (2)$$

for any $s, t \in [0, 1]$. Such map h is called *homotopy*. The equivalence class of paths under the homotopic relation is called a *homotopy class*. The homotopy class of a path γ is denoted as $[\gamma]$.

For example in Fig 1, paths p_2 and p_3 are homotopic because we can construct a homotopy between them in the free space. On the other hand, p_1 and p_2 are not homotopic because the obstacle prevent them from deforming into each other.

For a path γ , γ^{-1} denotes the path that follows γ in reverse. Given two paths γ_0, γ_1 such that $\gamma_0(1) = \gamma_1(0)$, we define the composition $\gamma_0 \circ \gamma_1$ as

$$\gamma_0 \circ \gamma_1(t) = \begin{cases} \gamma_0(2t) & 0 \leq t \leq 1/2, \\ \gamma_1(2t - 1) & 1/2 \leq t \leq 1. \end{cases} \quad (3)$$

Consider a point, $x_0 \in X$. The set of homotopy classes of closed loops with endpoints at x_0 (i.e., $\gamma : [0, 1] \rightarrow X$ such that $\gamma(0) = \gamma(1) = x_0$) forms a group under path composition. The identity element of this group is the homotopy class of the constant map to x_0 . The inverse element of $[\gamma]$ is $[\gamma^{-1}]$. When X is path-connected, this group is independent of the choice of x_0 up to isomorphism and denoted as $\pi_1(X)$. It is called the *fundamental group of X* (Hatcher 2002).

For example, when $X = \mathbb{R}^2$, the fundamental group is trivial because all paths with the same endpoints are homotopic. Figure 3 shows the simplest example for the nontrivial fundamental group where $X = \mathbb{R}^2 \setminus D$ and D is a disk. The homotopy class of a loop is determined by how many times it goes around the obstacle and in which direction. Furthermore, any homotopy class can be written as $[\gamma^n]$ with $n \in \mathbb{Z}$ and the fundamental group is isomorphic to \mathbb{Z} .

Note that, when X is path-connected, for any two points $x_0, x_1 \in X$, there exists a bijection between homotopy classes of paths from x_0 to x_1 and the fundamental group. Indeed, when we fix a path γ_0 from x_0 to x_1 , the homotopy class of any path γ from x_0 to x_1 is represented by $[\gamma_0^{-1} \circ \gamma] \in \pi_1(X)$.

3.2 Homotopy-Augmented Graph

Homotopy-aware search-based path planning is reduced to pathfinding on a *homotopy-augmented graph* (Bhattacharya and Ghrist 2018). Intuitively, the homotopy-augmented graph is a graph where homotopical information has been added to the vertices. A path in the original graph can be lifted to a path in the homotopy-augmented graph. For two paths with the same start and goal points in the original graph, the goal points of the lifted paths are the same if and only if the paths are homotopic. Therefore, finding multiple non-homotopic paths from a given start to a given goal is reduced to finding paths to multiple vertices of the homotopy-augmented graph that

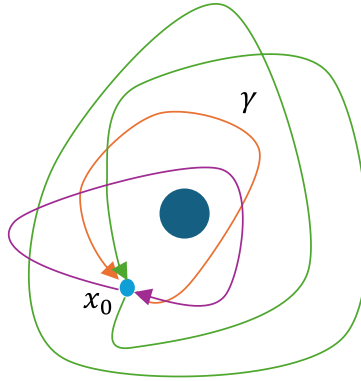


Fig. 3. Simplest example for the nontrivial fundamental group where there is a single obstacle (a disk) on the plane. A point x_0 is the base point and a loop γ from x_0 goes around the obstacle counterclockwise once. The loop that goes around the obstacle clockwise once is homotopic to γ^{-1} . The loop that goes around the obstacle counterclockwise twice is homotopic to γ^2 , which traces γ twice.

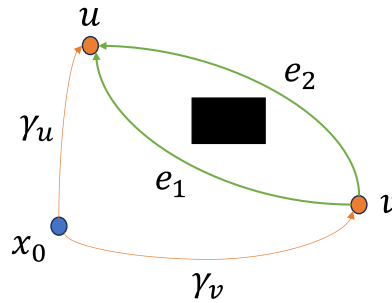


Fig. 4. Example of construction of homotopy-augmented graph. The vertices are labeled u and v and edges are labeled e_1 and e_2 . The base point x_0 and the paths γ_u and γ_v are selected to calculate the homotopy classes. The homotopical difference between edges e_1 and e_2 is represented by different elements $[\gamma_v \circ e_1 \circ \gamma_u^{-1}]$ and $[\gamma_v \circ e_2 \circ \gamma_u^{-1}]$ in $\pi_1(X)$.

correspond to the goal vertex with different homotopical information. Mathematically, the homotopy-augmented graph can be considered as a lift of the original graph into the universal covering space. However, we do not define this in this paper.

Let X be a path-connected space, and let $G = (V, E)$ be a discrete graph (roadmap) on X . The homotopy-augmented graph $G_h = (V_h, E_h)$ is constructed as follows⁵. We fix a base point $x_0 \in X$ for the fundamental group. We also assume that one homotopy class $[\gamma_v]$ of paths from x_0 to each vertex $v \in V$ is fixed. The construction of G_h is then expressed as follows:

- $V_h := V \times \pi_1(X)$, where $\pi_1(X)$ is the fundamental group of X with respect to the chosen base point x_0 .
- For each edge $e \in E$ from vertex $v \in V$ to vertex $u \in V$, and for each element $\alpha \in \pi_1(X)$, E_h contains an edge from (v, α) to $(u, \alpha h(e))$, where $h(e) := [\gamma_v \circ e \circ \gamma_u^{-1}]$. Here, γ_u and γ_v are paths in homotopy classes $[\gamma_u]$ and $[\gamma_v]$, respectively.

⁵Our description is generalized from that in the original paper to be independent of how homotopy is calculated.

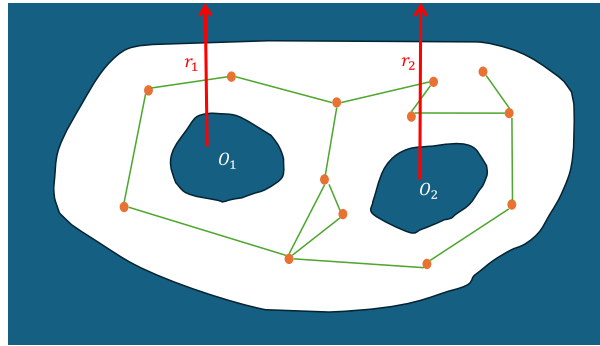


Fig. 5. Field with obstacles and roadmap on it. Rays from obstacles are used to calculate the homotopy classes of paths on the free space.

Figure 4 is a simple example. While $\gamma_v \circ e_1 \circ \gamma_u^{-1}$ is homotopic with the constant function, $\gamma_v \circ e_2 \circ \gamma_u^{-1}$ is not due to the existence of the obstacle. Thus, $h(e_1)$ is the identity element, and $h(e_2)$ is the element corresponding to one counterclockwise turn around the obstacle. In the homotopy-augmented graph, two edges above e_1 and e_2 with the same source have different targets.

Figure 5 is another example. We fix rays which do not pass through any vertex of the roadmap from each obstacle. We fix an arbitrary point that does not lie on any ray as the base point x_0 and take a path that does not intersect any ray as γ_v for a vertex v . Then, for an edge e , the element $h(e)$ is trivial when e does not intersect any ray. Let ρ_i be the element of the fundamental group corresponding to the loop that goes around the obstacles O_i without intersecting another ray, for $i = 1, 2$. Then, $h(e)$ is ρ_i or ρ_i^{-1} when e intersects a ray r_i only once. The sign depends on the direction of intersection. This approach is equivalent to the classical method for homotopy-aware single-agent path planning on the plane with obstacles (Grigoriev and Slissenko 1998; Tovar et al. 2010). In this method, during path planning, we detect crossing rays to calculate the homotopy class. We construct a string called a *word*, which is explained more generally in the next subsection, by adding a generator ρ_i or its inverse ρ_i^{-1} whenever traversing a ray r_i . This word represents the homotopy class. A consecutive pair of a generator and its inverse in a word can be canceled. A word which does not contain such pairs is called *reduced*. A vertex of the homotopy-augmented graph consists of a vertex of the original graph and a reduced word, which represents a homotopy class uniquely.

3.3 Presentation of Group

To describe the braid group in the next subsection, we explain the notion of presentation of a group, which is a way to specify a group.

A subset S of a group G is called a *set of generators* of G if any element of G can be written as a finite product of elements of S and their inverses. An element of S is called a *generator*. A *word* is a string consisting of generators and their inverses, which represents an element of the group. A *relation* is a word which represents the identity element.

Intuitively, a *presentation* $\langle S|R \rangle$ of a group G consists of a set S of generators and a set R of relations which is sufficient to specify G . Formally, $\langle S|R \rangle$ is a presentation of G if the following conditions are satisfied.

- S is a set of generators.
- The kernel of the natural surjection from the free group F_S on S to G is the smallest normal subgroup of F_S containing R .

A relation $ab^{-1} \in R$ is sometimes denoted as $a = b$.

The decision problem of determining whether two given words represent the same element of the group is known as the *word problem* (Peifer 1997).

For example, the fundamental group for the example in Figure 5 has a presentation $\langle \rho_1, \rho_2 | \emptyset \rangle$. Since there exists no relation between ρ_1 and ρ_2 , the set of relations is empty. For another example, if a group G has a presentation $\langle a, b | aba^{-1}b^{-1} \rangle$, then G is isomorphic to $\mathbb{Z} \times \mathbb{Z}$ because any element is written as $a^n b^m$ uniquely with $n, m \in \mathbb{Z}$ uniquely.

3.4 Braid Group

For a domain \mathcal{D} , the *configuration space* $C_n(\mathcal{D})$ is defined as

$$C_n(\mathcal{D}) := \{(p_1, p_2, \dots, p_n) \in \mathcal{D}^n \mid p_i \neq p_j \text{ for all } i \neq j\}. \quad (4)$$

As the name suggests, this space is the configuration space for the multi-agent path planning of n agents without size in \mathcal{D} , because the excluded area corresponds to states in which some two agents occupy the same position.

The n -th symmetric group S_n naturally acts on $C_n(\mathcal{D})$ by permuting the indices. The *unlabeled configuration space* $\mathcal{UC}_n(\mathcal{D})$ is defined as the quotient of $C_n(\mathcal{D})$ by S_n . This space is the configuration space for unlabeled multi-agent path planning. In this problem setting, agent indexes are ignored, starts and goals do not correspond, and each agent may reach any goal, while exactly one agent must reach at each goal.

The fundamental group of $\mathcal{UC}_n(\mathbb{R}^2)$ is known as the n -th *braid group* B_n . The fundamental group of $C_n(\mathbb{R}^2)$, which is a subgroup of B_n , is known as the n -th *pure braid group* P_n . There exists a natural surjective group homomorphism from B_n to S_n , the kernel of which is P_n .

B_n has a presentation given by

$$\langle \sigma_1, \dots, \sigma_{n-1} \mid \sigma_i \sigma_j \sigma_i^{-1} \sigma_j^{-1}, \sigma_i \sigma_{i+1} \sigma_i \sigma_{i+1}^{-1} \sigma_i^{-1} \sigma_{i+1}^{-1} \rangle, \quad (5)$$

where $1 \leq i < n-1$, and $i+1 < j \leq n-1$ (Rolfsen 2010). See § 5.3 for the geometrical description of this presentation.

4 Problem Setting

In this section, we describe the setting for homotopy-aware path planning in planar domains.

We consider homotopy-aware multi-agent path planning with n agents in the domain $\mathcal{D} \subseteq \mathbb{R}^2$. As illustrated in Figure 6, we assume that \mathcal{D} has the form $D_0 \setminus (O_1 \cup \dots \cup O_r)$, where D_0 and O_1, \dots, O_r (obstacles) are bounded domains whose boundaries are simple closed curves and $\mathbb{R}^2 \setminus D_0, O_1, \dots, O_r$ neither intersect nor touch.⁶

We assume that a finite undirected planar graph $G = (V, E)$ lying on the interior of \mathcal{D} is given as a roadmap. Starts $s_1, \dots, s_n \in V$ and goals $g_1, \dots, g_n \in V$ are also given. Time is discretized into steps, and each agent either moves along an edge or remains at a vertex in each step. Namely, a single-agent plan for the i -th agent is a sequence $v_{i,0}, v_{i,1}, \dots, v_{i,T} \in V$ of vertices such that $v_{i,0} = s_i$, $v_{i,T} = g_i$, and $v_t = v_{t+1}$ or $(v_t, v_{t+1}) \in E$ for any t . A *solution* is a tuple of single-agent plans for all agents without any vertex or swapping conflicts, i.e., $v_{i,t} \neq v_{j,t}$ and $(v_{i,t}, v_{i,t+1}) \neq (v_{j,t+1}, v_{j,t})$ for all $i \neq j$ (Stern et al. 2019). The *sum of costs* of a solution is defined as the sum of the reaching times for all agents, which means

$$\sum_{i=1}^n \min\{t \mid v_{i,t} = g_i\}. \quad (6)$$

⁶Strictly speaking, for example, a domain in a grid with obstacles touching at corners does not satisfy this assumption. However, such obstacles can be considered as one obstacle by widening them slightly at the corners.

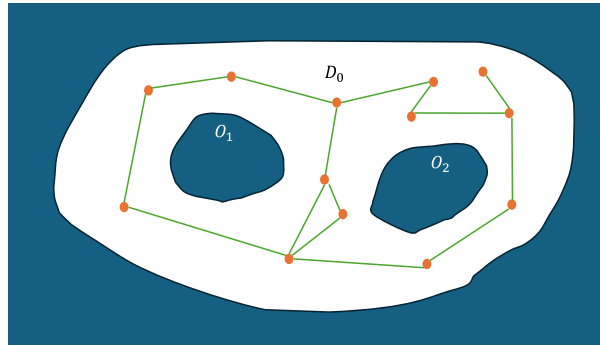


Fig. 6. Example of planar domain \mathcal{D} satisfying the assumption of the problem setting and roadmap G over \mathcal{D} . Here, $\mathcal{D} = \mathcal{D}_0 \setminus (O_1 \cup O_2)$, where \mathcal{D}_0 is the region enclosed by the outer boundary and O_1 and O_2 are obstacles.

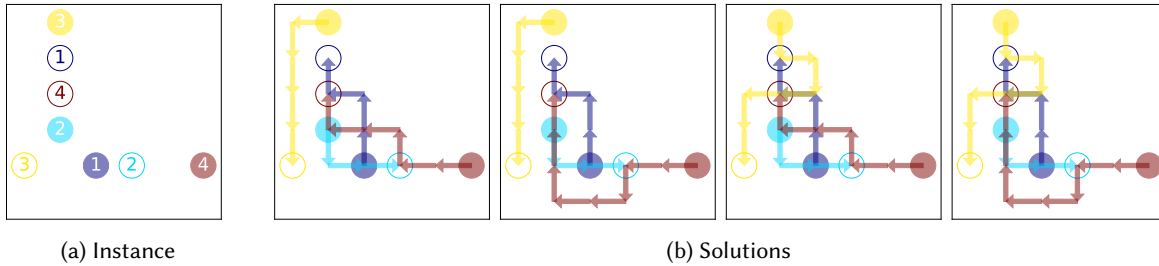


Fig. 7. Examples of Instance of MAPF on grid and its homotopically distinct solutions. The filled circles are for the starts, and the unfilled circles with the same indexes are for the corresponding goals. The moves of agents are represented by arrows. In these solutions, no agents stay at the same vertex except at their goals. The displayed four solutions belong to different homotopy classes. More specifically, we can classify them based on which side the third agent avoids the first agent and which side the fourth agent avoids the second agent.

A solution determines a path in the configuration space $C_n(\mathcal{D})$. Two solution are called *homotopic* if their corresponding paths are homotopic. When $K > 0$ is given, We want to find K mutually non-homotopic solutions with small sums of costs.

Figure 7 shows an instance and its homotopically distinct solutions in a grid world.

As stated in § 3, the problem can be reduced to pathfinding on the homotopy-augmented graph in $C_n(\mathcal{D})$.

Remark 1. The problem of finding K mutually non-homotopic paths with minimum lengths is called *K -shortest non-homotopic path planning* (Bhattacharya and Ghrist 2018; Yang et al. 2022). Since our priority in this paper is scalability, outputs of our method are not necessarily optimal. See also Remark 6.

Remark 2. For simplicity, we consider the case where G has edges of the equal cost with no crossing, and where only vertex and swapping conflicts are forbidden. Our framework works even when time is continuous or collision conditions are complex (Kasaura, Nishimura, et al. 2022), because these conditions only affect the construction of the timed graph in Line 8 in Algorithm 1. However, such collision conditions are ignored in the homotopy calculations. See also the next remark.

Remark 3. Since the homotopy classes are defined in the configuration space, intermediate paths do not need to correspond to any solutions in G . The agents are considered as points during continuous deformation of solutions.

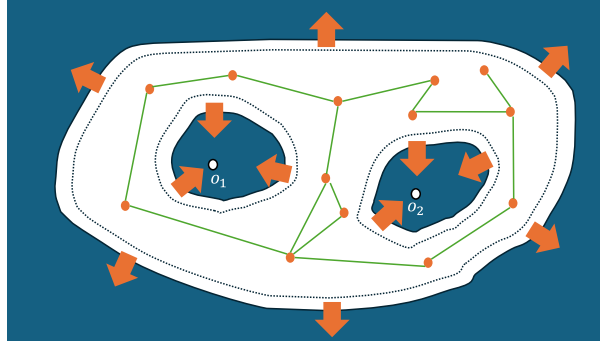


Fig. 8. Deformation of $\mathcal{D} = \mathcal{D}_0 \setminus (O_1 \cup O_2)$ to $\mathbb{R}^2 \setminus \{o_1, o_2\}$, where $o_1 \in O_1$ and $o_2 \in O_2$. Regions near obstacles expand to collapse obstacles to points. A region near the outer boundary expands to the point at infinity. Points in the region enclosed by dotted curves, which includes roadmap G , are fixed during deformation.

Note that this simplification only affects homotopies; sizes of agents are considered in pathfinding. Ignoring the sizes of agents does not matter in cases with no obstacles because the agents can be far enough apart from each other when deforming paths to other ones. On the other hand, when there exist obstacles, two paths that are actually homotopically distinct when both sizes of agents and obstacles are taken into account may be considered as homotopic. See also § 7. However, the fact that our homotopical classification is not complete does not mean that it is useless in such cases.

5 Method

In this section, we describe our method to generate multiple solutions with different homotopies.

The entire algorithm, which is provided in § 5.5, is a combination of revised prioritized planning and our framework for calculating homotopy classes while planning. In the first four subsections, we describe the framework. For the setting where agents are not labeled and there are no obstacles, a classical method to construct words representing homotopy classes is known, which is described in § 5.3. In § 5.1 and § 5.2, we reduce our problem setting to this case. Since constructed words themselves are not suitable for comparison, a method to represent them by tuple of integers, which is called Dynnikov coordinates, is explained in § 5.4. In § 5.6, we prove the completeness of our algorithm under specific assumptions.

5.1 Reduction to Obstacle-Free Case

We take one representative element $o_i \in O_i$ for each obstacle. Obviously, o_1, \dots, o_r are different from each other. By taking sufficiently small neighborhoods of boundaries and transforming them, as illustrated in Figure 8, we can deform \mathcal{D} to $\mathbb{R}^2 \setminus \{o_1, \dots, o_r\}$ without changing the roadmap. For a rigorous proof, the Jordan–Schönflies theorem (Thomassen 1992) can be used. Thus, it is enough to calculate homotopies on $\mathbb{R}^2 \setminus \{o_1, \dots, o_r\}$. Note that, since the roadmap is not changed, the found paths are lying on the original \mathcal{D} .

We reduce the calculation of homotopy in $C_n(\mathbb{R}^2 \setminus \{o_1, \dots, o_r\})$ to that in $C_{r+n}(\mathbb{R}^2)$. Intuitively, we consider the obstacles as additional agents staying at the same positions. While it is non-trivial that this reduction preserves homotopy classes because obstacles cannot move when deforming paths, the following proposition guarantees it.

PROPOSITION 4. *The map*

$$\pi_1(C_n(\mathbb{R}^2 \setminus \{o_1, \dots, o_r\})) \rightarrow \pi_1(C_{r+n}(\mathbb{R}^2)) \quad (7)$$

induced by the embedding

$$\begin{array}{ccc} C_n(\mathbb{R}^2 \setminus \{o_1, \dots, o_r\}) & \hookrightarrow & C_{r+n}(\mathbb{R}^2) \\ \downarrow \psi & & \downarrow \psi \\ (p_1, \dots, p_n) & \mapsto & (o_1, \dots, o_r, p_1, \dots, p_n) \end{array} \quad (8)$$

is injective.

See Appendix B for the proof.

In the following three subsections, we replace $r + n$ with n and consider $C_n(\mathbb{R}^2)$ for simplicity. We also write simply C_n and \mathcal{UC}_n instead of $C_n(\mathbb{R}^2)$ and $\mathcal{UC}_n(\mathbb{R}^2)$.

5.2 Reduction to Unlabeled Case

As mentioned earlier, in labeled multi-agent path planning, we can calculate homotopy classes as unlabeled multi-agent path planning. To formalize this reduction, let $G^n = (V^n, E^n)$ be the n -times direct product of G minus collision parts, which is a graph on the configuration space C_n . Instead of directly applying the construction in § 3.2 to C_n , we construct a graph (V_h^n, E_h^n) as follows.

- $V_h^n := V^n \times \pi_1(\mathcal{UC}_n) = V^n \times B_n$.
- For each edge $e \in E^n$ from vertex $v \in V^n$ to vertex $u \in V^n$, and for each element $\alpha \in B_n$, E_h^n contains an edge from (v, α) to $(u, \alpha h(\bar{e}))$, where \bar{e} is the projection of e to \mathcal{UC}_n by natural surjection, and $h(\bar{e})$ is calculated using the method described in § 5.3.

It is important to note that, for a specific multi-agent path planning task, it is not necessary to construct the entire graph as it is not connected. Instead, we can focus on the relevant connected components of the graph. For example, for any vertex $v \in V^n$ and elements $\alpha, \beta \in B_n$, the connected components of (v, α) and (v, β) are different unless $\alpha^{-1}\beta \in P_n$.

With our approach, we represent the homotopy classes of solutions for a specific instance of the labeled multi-agent path planning problem by labeling them with elements in a coset of the pure braid group, instead of those in the pure braid group. The choice of coset depends on the relative positions of the agents' start and goal locations. For a detailed explanation of why we do not use the pure braid group, see Appendix A.

5.3 Word Construction

The construction of words for \mathcal{UC}_n is classically known (Fox and Neuwirth 1962).

We fix a coordinate x, y for \mathbb{R}^2 . Let $p_1 = (x_1, y_1), \dots, p_n = (x_n, y_n)$ be the induced coordinates of C_n . We define:

$$\lambda_{2n} := \{x_1 < x_2 < \dots < x_n\}, \quad (9)$$

$$\lambda_i^{2n-1} := \{x_1 < \dots < x_i = x_{i+1} < \dots < x_n, y_i < y_{i+1}\}, \quad (10)$$

for $1 \leq i < n$. It is easy to see that the map $\lambda^{2n} \cup \bigcup_i \lambda_i^{2n-1} \hookrightarrow C_n \rightarrow \mathcal{UC}_n$ is injective. We identify $\lambda^{2n}, \lambda_i^{2n-1}$ with their images. $\mathcal{UC}_n \setminus \lambda^{2n}$ is $(2n - 1)$ -dimensional and $\mathcal{UC}_n \setminus \lambda^{2n} \setminus \bigcup_i \lambda_i^{2n-1}$ is $(2n - 2)$ -dimensional.

For simplicity, we assume that all $v \in V^n$ lie on λ^{2n} . We also choose a base point x_0 of the fundamental group within λ^{2n} . To make the homotopy-augmented graph explicit, we choose the standard path connecting x_0 and vertices of V^n in λ^{2n} . The generator σ_i of $\pi_1(\mathcal{UC}_n) = B_n$ is represented by a loop in $\lambda^{2n} \cup \lambda_i^{2n-1}$ that traverses λ_i^{2n-1} exactly once with the direction from region $\{y_i < y_{i+1}\}$ to region $\{y_i > y_{i+1}\}$. Consequently, for an edge e on \mathcal{UC}_n , the word representing the corresponding braid $h(e)$ is constructed by detecting the traverses of e with $\lambda_1^{2n-1}, \dots, \lambda_{n-1}^{2n-1}$. By examining the intersections of the boundaries of $\lambda_1^{2n-1}, \dots, \lambda_{n-1}^{2n-1}$, we obtain the relations $\sigma_i \sigma_j = \sigma_j \sigma_i$ for $i + 1 < j$ and $\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$. Detailed derivations of these relations are omitted for brevity.

Intuitively, this construction can be described as follows. We number the agents in ascending order of their x -coordinates. σ_i corresponds to the counterclockwise swap (Figure 2a) of the agent with number i and the

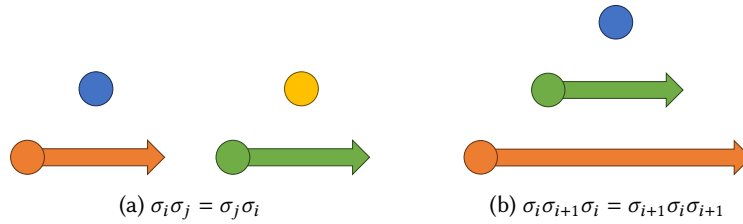


Fig. 9. Figures illustrating relations between the generators. In (a), the left circles represent the agents with numbers i and $(i + 1)$, and the right circles represent the agents with numbers j and $(j + 1)$, where $j > i + 1$. In (b), the bottom, middle, and top circles represent the agents with numbers i , $(i + 1)$, and $(i + 2)$, respectively. Arrows represent movements. The homotopy being independent of the order of moves corresponds to the relations.

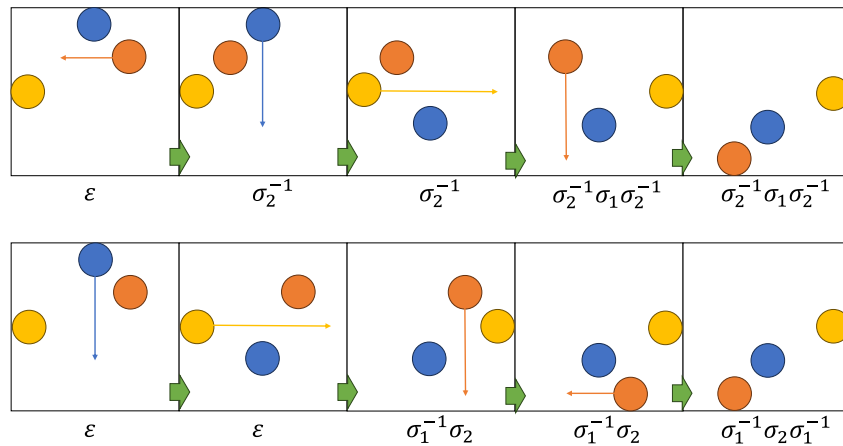


Fig. 10. Two examples of braid construction. From left to right, generators are added or not added to the displayed braid word as an agent moves. In the top example, the agents with numbers 2 and 3 swap clockwise at the first block, then the agents with numbers 1 and 2 swap counterclockwise and the agents with numbers 2 and 3 swap clockwise at the third block. In the example below, the agents with numbers 1 and 2 swap clockwise and the agents with numbers 2 and 3 swap counterclockwise at the second block; then, the agents with numbers 1 and 2 swap clockwise at the fourth block.

agent with number $(i + 1)$, and σ_i^{-1} corresponds to their clockwise swap (Figure 2b). The word for an edge is constructed by detecting the swaps of agent indexes. The graphical explanation for the relations is shown in Figure 9. The key observation is that the order in which the agents move does not affect the homotopy class. Therefore, we have two relations: (a) $\sigma_i \sigma_j = \sigma_j \sigma_i$ and (b) $\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$.

Figure 10 shows two examples of word construction.

Remark 5. The word construction depends on the choice of the coordinates x and y , because the choice of generators depends on them. However, since homotopy itself is independent of the coordinates, if braids for two paths with the same endpoints calculated by using some coordinates are the same, their braids always coincide even when other coordinates are used.

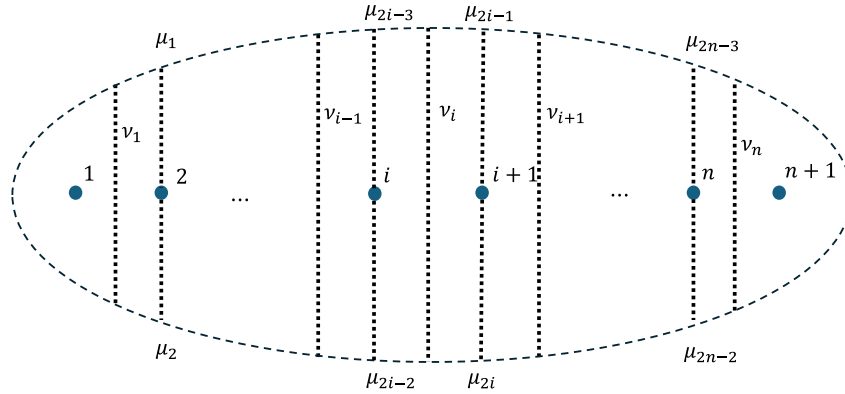


Fig. 11. Plane with $(n + 1)$ holes to calculate Dynnikov coordinates. For representation of multicurves on it by integers, the dotted lines or half lines are used to count the numbers of times multicurves intersect them.

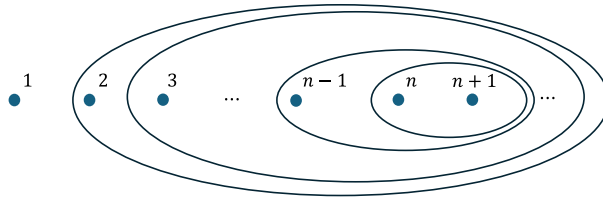


Fig. 12. Initial multicurve to solve the word problem for braids with Dynnikov coordinates.

5.4 Dynnikov Coordinates

We use Dynnikov coordinates to uniquely represent braids. In brief, we represent the braid group as actions to tuples of integers, which can be calculated only by using addition, subtraction, maximum, and minimum. We follow the description in [Thiffeault \(2022\)](#).

Intuitively, Dynnikov coordinates represent a multicurve on a plane with aligned $(n + 1)$ holes.⁷ Here, “multicurve” means a disjoint union of simple closed curves, any of which encircles at least two holes but not all of them. For an example, see the example calculation at the end of this subsection. Figure 11 illustrates a plane with holes, where symbols attached to lines or half lines indicate the number of times the multicurve intersects them. Holes are numbered from left to right. For $2 \leq i \leq n$, μ_{2i-3} and μ_{2i-2} correspond to the half lines extending up and down from the i -th hole, respectively. For $1 \leq i \leq n$, v_i corresponds to the vertical line between the i -th and $(i + 1)$ -th holes. When counting the numbers of intersections, we take the minimum ones, allowing for homotopical deformation of curves.

For any $1 \leq i \leq n - 1$, let

$$a_i := \frac{\mu_{2i} - \mu_{2i-1}}{2},$$

$$b_i := \frac{v_i - v_{i+1}}{2}.$$

⁷While n holes are enough to define the action by B_n , this action is not faithful.

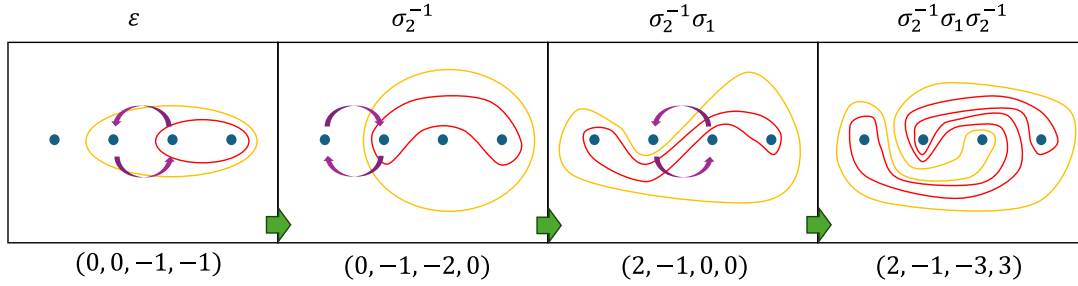


Fig. 13. Action of braid $\sigma_2^{-1}\sigma_1\sigma_2^{-1}$ to multicurve and corresponding Dynnikov coordinates. The top shows braids, the middle shows multicurves after these braids acted on u , and the bottom shows the corresponding Dynnikov coordinates. The Dynnikov coordinates corresponding to $\sigma_2^{-1}\sigma_1\sigma_2^{-1}$ are $(2, -1, -3, 3)$.

Then, $(a_1, \dots, a_{n-1}, b_1, \dots, b_{n-1})$ gives a bijection between homotopy classes of multicurves and $\mathbb{Z}^{2n-2} \setminus \{0\}$, that is *Dynnikov coordinates* (Hall and Yurttas 2009).

The braid group B_{n+1} acts on the homotopy classes of multicurves by moving holes. For an example, again, see the example calculation at the end of this subsection. Here, we are only interested in the action of B_n as a subgroup of B_{n+1} . The action of B_n can be written using Dynnikov coordinates as follows. For $x \in \mathbb{Z}$, we write $x^+ := \max\{x, 0\}$ and $x^- := \min\{x, 0\}$. For $i = 1, \dots, n-1$ and $e = \pm 1$, we can calculate $(a'_1, \dots, a'_{n-1}, b'_1, \dots, b'_{n-1}) := (a_1, \dots, a_{n-1}, b_1, \dots, b_{n-1}) \cdot \sigma_i^e$ as

$$(a'_k, b'_k) := \begin{cases} (-b_1 + (a_1 + b_1^+)^+, a_1 + b_1^+) & \text{for } i = k = 1, e = +1, \\ (b_1 - (b_1^+ - a_1)^+, b_1^+ - a_1) & \text{for } i = k = 1, e = -1, \\ (a_{i-1} - b_{i-1}^+ - (b_i^+ + c)^+, b_i + c^-) & \text{for } i > 1, k = i - 1, e = +1, \\ (a_{i-1} + b_{i-1}^+ + (b_i^+ - d)^+, b_i - d^+) & \text{for } i > 1, k = i - 1, e = -1, \\ (a_i - b_i^- - (b_{i-1}^- - c)^-, b_{i-1} - c^-) & \text{for } i > 1, k = i, e = +1, \\ (a_i + b_i^- + (b_{i-1}^- + d)^-, b_{i-1} + d^+) & \text{for } i > 1, k = i, e = -1, \\ (a_k, b_k) & \text{for } k \neq i - 1, i, \end{cases} \quad (11)$$

where $c := a_{i-1} - a_i - b_i^+ + b_{i-1}^-$ and $d := a_{i-1} - a_i + b_i^+ - b_{i-1}^-$ (Thiffeault 2022).

Moreover, where $u := (a_1 = 0, \dots, a_{n-1} = 0, b_1 = -1, \dots, b_{n-1} = -1) \in \mathbb{Z}^{2n-2}$, which corresponds to the multicurves in Figure 12, for any $\alpha, \beta \in B_n$, $u \cdot \alpha = u \cdot \beta$ if and only if $\alpha = \beta$ (Dynnikov 2002; Thiffeault 2022). Thus, we can represent an element α in B_n by $u \cdot \alpha \in \mathbb{Z}^{2n-2} \setminus \{0\}$.

Figure 13 shows examples of the action of the braid group and Dynnikov coordinates when $n = 3$.

Since the values of the coordinates can be large, multiple-precision arithmetic is necessary for its implementation. The logarithmic magnitude of the coordinates is $O(l)$, where l is the length of the braid word, and computational cost for one update (11) is proportional to the logarithmic magnitude of the coordinates. Therefore, the time complexity of comparing braid words of length l is $O(l^2)$, which is the best complexity among known algorithms. Furthermore, since this calculation only requires simple arithmetic operations, it can be highly efficient in practice (Dehornoy 2008). See also § 6.3.5.

5.5 Revised Prioritized Planning

In this and the following subsections, we revert to the notation used prior to the end of § 5.1. Thus, n denotes the number of agents and r denotes the number of obstacles.

Algorithm 1 Homotopy-Aware RPP

```

1: Input a graph  $G = (V, E)$  and starts and goals  $(s_1, g_1), \dots, (s_n, g_n)$ .
2: Input the desired number  $K$  of solutions
3:  $plans \leftarrow \{\emptyset\}$ 
4: for  $i = 1, \dots, n$  do
5:    $Open, Closed \leftarrow \emptyset, \emptyset$ 
6:   Initialize a map  $D$ 
7:   for  $plan \in plans$  do
8:     Construct timed graph  $G_{plan} = (V_{plan}, E_{plan})$ 
9:     Insert  $(plan, s_{plan}, \varepsilon)$  to  $Open$ 
10:     $D[plan, s_{plan}, \varepsilon] \leftarrow 0$ 
11:   end for
12:    $newplans \leftarrow \emptyset$ 
13:   while  $Open \neq \emptyset$  do
14:     Pop  $(plan, v, w)$  from  $Open$ 
15:     Insert  $(plan, v, w)$  to  $Closed$ 
16:     if  $v = g_{plan}$  then
17:       Reconstruct path  $p$  of agent  $i$ 
18:       Insert  $plan \cup p$  to  $newplans$ 
19:       if  $|newplan| \geq K$  then
20:         break
21:       end if
22:     end if
23:     for edge  $e \in E_{plan}$  from  $v$  do
24:        $v', w', d' \leftarrow \text{target}(e), \text{NextBraid}(plan, e, w), D[plan, v, w] + \text{length}(e)$ 
25:       if  $(plan, v', w') \notin Open \cup Closed$  then
26:          $D[plan, v', w'] \leftarrow d'$ 
27:         Insert  $(plan, v', w')$  to  $Open$ 
28:       else if  $D[plan, v', w'] > d'$  then
29:          $D[plan, v', w'] \leftarrow d'$ 
30:       end if
31:     end for
32:   end while
33:    $plans \leftarrow newplans$ 
34: end for
35: return  $plans$ 

```

We adopt *revised prioritized planning* (RPP) (Čáp, Novák, et al. 2015) for pathfinding. As classical prioritized planning, we fix a predetermined priority order for the agents and proceed to plan their paths one by one, avoiding collisions with the paths of previously planned agents. In RPP, in addition, the start positions of agents that have not yet been planned are also avoided, ensuring the completeness of the planning process under certain assumptions.

Our method differs from conventional RPPs in the following ways: First, we maintain multiple plans for higher-priority agents with different homotopies while planning paths one by one. Planning for the i -th agent

is performed on the homotopy-augmented graph for the first i agents in $C_i(\mathcal{D})$. For multiple plans for already planned agents, the planning for the next agent is performed on different graphs, but they are done in parallel.

Algorithm 1 presents the pseudocode of our homotopy-aware version of RPP, which generates homotopically distinct solutions. The notation ε denotes the Dynnikov coordinates corresponding to the empty braid. At line 8, we construct a graph G_{plan} , which is created by adding a time dimension to G and removing parts colliding with $plan$, the same as Silver (2005).⁸ In this construction, we also remove s_{i+1}, \dots, s_n from G_{plan} . (Strictly speaking, we allow an agent to reach one of such positions if the position is its goal, since it cannot finish otherwise.) In the following lines, s_{plan} and g_{plan} denote the vertices corresponding to the start, which is s_i with time 0, and the goal, which is g_i with a late enough time, respectively. A vertex of the homotopy-augmented graph for $plan$ is represented by a pair (v, w) of a vertex v of G_{plan} and a braid $w \in B_{r+i}$.⁹ At line 14, the popped element $(plan, v, w)$ is selected as that with the minimum $\text{cost}(plan) + D[plan, v, w] + h(v, g_{plan})$, where h is the heuristic function.

The function $\text{NextBraid}(plan, e, w)$ at line 24 returns the Dynnikov coordinates updated from w after the agent i moves along e and the agents from 1 to $i - 1$ move in accordance with $plan$. It is calculated as follows. First, as explained in § 5.3, the swaps of the order of the agents' x -coordinates by these moves are enumerated. Second, for each swap, the Dynnikov coordinates are updated by (11) in order. After calculating the next braid w' , as in the conventional A* algorithm, the target vertex (v', w') is checked to see if it has been visited. If so, it is added. Otherwise, its distance is updated or nothing happens.

For simplicity, we omit the details on path reconstruction.

Remark 6. While we adopt the RPP approach for scalability in this paper, we can also solve the K -shortest non-homotopic path planning by A* searching on the homotopy-augmented graph in $C_n(D)$.

5.6 Completeness

As RPP in the classical case, we can prove the completeness of the algorithm under some specific condition for problem instances. Note that this does not mean that our algorithm is practical only for such cases.

Before the completeness proposition, we state a lemma used in the proof.

LEMMA 7. For any $1 \leq k \leq n$, let $F_k : P_n \rightarrow P_{n-1}$ be the projection forgetting the k -th point. For $1 \leq i < j \leq n$, let

$$a_{i,j} := \sigma_{j-1}\sigma_{j-2}\cdots\sigma_{i+1}\sigma_i^2\sigma_{i+1}^{-1}\cdots\sigma_{j-2}^{-1}\sigma_{j-1}^{-1} = \sigma_i^{-1}\sigma_{i+1}^{-1}\cdots\sigma_{j-2}^{-1}\sigma_{j-1}^2\sigma_{j-2}\cdots\sigma_{i+1}\sigma_i \in P_n. \quad (12)$$

Then, the kernel of F_k is generated by $a_{1,k}, a_{2,k}, \dots, a_{k-1,k}$ and $a_{k,k+1}, a_{k,k+2}, \dots, a_{k,n}$.

PROOF. This is true for $k = n$ (Rolfsen 2010). For arbitrary k , let $b_k := \sigma_{n-1}\sigma_{n-2}\cdots\sigma_k \in B_n$. Since P_n is a normal subgroup, the conjugation by b_k maps P_n to P_n . Furthermore, $F_k(\alpha) = F_n(b_k\alpha b_k^{-1})$. Thus, the kernel of F_k is generated by $b_k^{-1}a_{1,n}b_k, \dots, b_k^{-1}a_{n-1,n}b_k$, which can be calculated as

$$b_k^{-1}a_{i,n}b_k = \begin{cases} a_{i,k} & \text{if } i < k \\ a_{k,i+1} & \text{if } i \geq k. \end{cases} \quad (13)$$

□

Intuitively, the following proposition says that our algorithm is complete when the obstacles and start and goal positions are separated enough with respect to the size of the agents.

PROPOSITION 8. We assume the following conditions:

⁸If continuous time is considered, you can use the method in Phillips and Likhachev (2011).

⁹Strictly speaking, the positions of the first $(i - 1)$ agents are also contained in a vertex of the homotopy-augmented graph in $C_i(\mathcal{D})$. However, since these positions are determined by $plan$, they are omitted.

- An agent moving along the boundary of \mathcal{D} never collides with obstacles or other agents staying at $s_1, \dots, s_n, g_1, \dots, g_n$, or s_{i+1}, \dots, s_n .
- For any $1 \leq i \leq n$, there exists a loop around g_i in \mathcal{D} such that it does not enclose any obstacle, other goal position, or s_j with $j > i$, and an agent moving along it never collides with obstacles or other agents staying at any goal position or s_j with $j > i$.
- For any $1 \leq i \leq n$, there exists a loop around s_i such that it does not enclose any obstacle, other start position, or g_j with $j < i$, and an agent moving along it never collides with obstacles or other agents staying at any start position or g_j with $j < i$.

Then, when the roadmap G is dense enough, for any homotopy class of solutions, Algorithm 1 provides a solution belonging to it for sufficiently large $K \gg 0$.

PROOF. We consider a problem instance $(s_1, g_1), \dots, (s_n, g_n)$ satisfying the assumption and a braid $\alpha \in B_{r+n}$ corresponding to one of its solutions. For $0 \leq i \leq n$, let $\alpha_i \in B_{r+i}$ be the braid corresponding to the paths for the first i agents in α . We prove that our method generates a solution p_i in the homotopy class corresponding to α_i for sufficiently large $K \gg 0$ by induction for i . When $i = 0$, this is clear because no agent is considered, and α_0 is the unit.

For $i \geq 1$, we assume that the plan p_{i-1} with the homotopy corresponding to α_{i-1} is contained in plans in Algorithm 1. Any reachable vertex in the homotopy-augmented graph can be found by repeating loop of Line 13 a sufficient number of times, since the number of vertices in the homotopy-augmented graph at distances less a given value is finite. Therefore, it suffices to show that there exists a plan p_i with the homotopy corresponding to α_i on the basis of p_{i-1} .

Under the assumption of the problem instance, the i -th agent can reach its goal after agents from 1 to $i - 1$ have finished while avoiding the goal positions g_1, \dots, g_{i-1} and the start positions s_{i+1}, \dots, s_n . Therefore, there exists a plan p'_i for the first i agents on the basis of p_{i-1} . Let $\beta_i \in B_{r+i}$ be the braid corresponding to p'_i .

To construct the desired plan, we need to add the moves of the i -th agent after p'_i . These moves must correspond to the braid $\beta_i^{-1}\alpha_i \in P_{r+i}$. Let c_1, \dots, c_{r+i} be $o_1, \dots, o_k, g_1, \dots, g_i$ sorted by their x -coordinates, and suppose $g_i = c_k$ ($1 \leq k \leq r + i$). Since α_i and β_i coincide when the i -th agent is forgotten, $\beta_i^{-1}\alpha_i$ is in the kernel of F_k , which is generated by $a_{1,k}, \dots, a_{k-1,k}, a_{k,k+1}, \dots, a_{k,r+i}$ as previously proved.

Thus, it is sufficient to demonstrate that, after all i agents have reached their goals, the i -th agent can move to construct any of $a_{1,k}, \dots, a_{k-1,k}, a_{k,k+1}, \dots, a_{k,r+i}$ and their inverses, while avoiding g_1, \dots, g_{i-1} and s_{i+1}, \dots, s_n .

For $1 \leq j < k$, the agent first moves up (in the positive direction of the y -axis) avoiding obstacles, g_1, \dots, g_{i-1} , and s_{i+1}, \dots, s_n by moving partially along the boundaries or the assumed loops. When avoiding the obstacle O_l , if the x coordinate of o_l is smaller than that of c_k , it avoids to the right (the positive side of the x -axis); otherwise, it avoids to the left. This is the same for g_1, \dots, g_{i-1} . Then, the agent moves to the left until it reaches the same x -coordinate as c_j along the outside boundary and moves down to c_j avoiding obstacles and other start or goal positions. When going down as well as going up, the agents avoid obstacles or goal positions represented by c_l to the right (from our point of view) if the x -coordinate of c_l is smaller than that of c_j ; otherwise, it avoids to the left. It circumvents c_j counterclockwise or clockwise and returns to c_k using the same path. The braid for such a loop is given by:

$$\sigma_{k-1} \cdots \sigma_{j+1} \sigma_j^{\pm 2} \sigma_{j+1}^{-1} \cdots \sigma_{k-1}^{-1} = a_{j,k}^{\pm 1} \quad (14)$$

where the sign depends on the direction for moving around c_j . Similarly, for $k < j \leq n$, the agent follows the path described above in reverse order to c_j , around c_j , and back. The braid is given by:

$$\sigma_k^{-1} \cdots \sigma_{j-2}^{-1} \sigma_{j-1}^{\pm 2} \sigma_{j-2} \cdots \sigma_k = a_{k,j}^{\pm 1} \quad (15)$$

Figure 14 illustrates examples of such paths.

When the roadmap is dense enough, these paths can be approximated by paths on the graph. \square

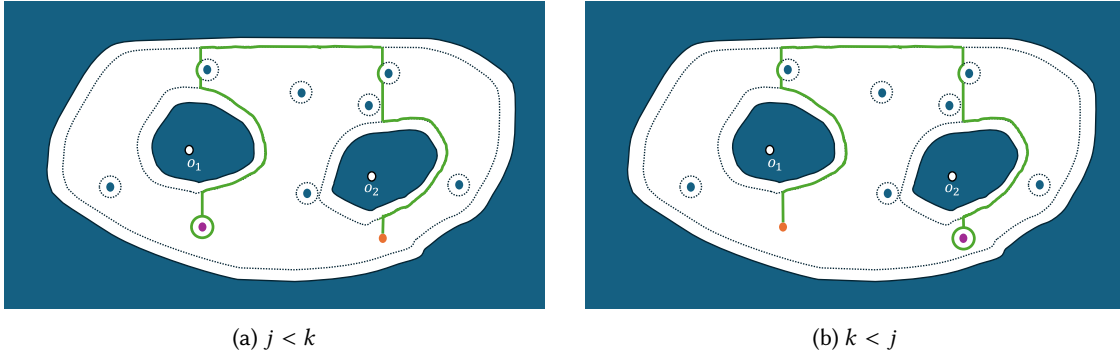


Fig. 14. Paths with homotopy corresponding to $a_{j,k}, a_{k,j}$ or its inverses. The circle enclosed by the path means $g_i = c_j$ and the circle at the endpoint of the path means c_k . Other circles mean other c_1, \dots, c_i or s_{i+1}, \dots, s_n .

In particular, in the case of the grid, by the construction of paths in the proof, the following corollary holds. To facilitate the word construction method in grid environments, we introduce a virtual slightly inclined x -axis, such that for any two grid cells (i, j) and (k, l) , cell (i, j) has a smaller x -coordinate than cell (k, l) if and only if $i < k$ or $i = k, j < l$.

COROLLARY 9. *We consider cases where \mathcal{D} is a region composed of grids, and G is a four-connective grid graph. We assume the following conditions:*

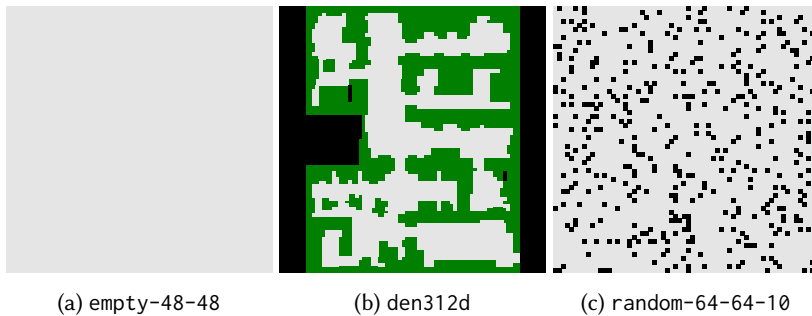
- No start grid or goal grid is adjacent to obstacle grids vertically, horizontally, or diagonally.
- No two start grids or two goal grids are adjacent vertically, horizontally, or diagonally.
- For any $1 \leq i < j \leq n$, g_i and s_j are not adjacent vertically, horizontally, or diagonally.

Then, for any homotopy class of solutions, Algorithm 1 generates a solution belonging to it for sufficiently large $K \gg 0$.

Remark 10. Proposition 8 states that, if the assumptions are met, then no homotopy classes are excluded in principle. However, the value of K required to generate a solution belonging to a specific homotopy class may be large. To find a solution with the homotopy corresponding to a given braid, Algorithm 1 can be modified as follows. Let $\alpha \in B_{r+n}$ be the given braid and let $\alpha_i \in B_{r+i}$ be the image of α under the projection forgetting all agents after the i -th one. Unlike Algorithm 1, we search for only one plan when planning for each agent, as in conventional RPP. For the i -th agent, we search for a path to the vertex (g_{plan}, α_i) in the homotopy-augmented graph. From the proof of Proposition 8, it is derived that, under the same assumptions as in the Proposition 8, this algorithm can find a desired solution.

6 Experiments

We conducted two experiments. In the first experiment, we measured the runtime of our method to assess its scalability, comparing it with that of a method using the Dehornoy order and the handle reduction algorithm (Dehornoy 1997) instead of Dynnikov coordinates. In the second experiment, our focus was to demonstrate the effectiveness of generating homotopically distinct coarse solutions for planning low-cost trajectories. We demonstrate how our approach can lead to improved trajectories by comparing the results obtained after optimization. The codes used for these experiments are available at <https://github.com/omron-sinix/homotopy-aware-MAPP>.



(a) empty-48-48 (b) den312d (c) random-64-64-10

Fig. 15. Maps from Moving AI Benchmark for evaluation of runtime.

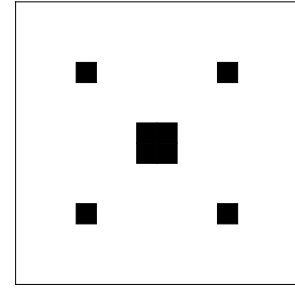


Fig. 16. Grid map with five obstacles for optimization experiment.

6.1 Implementation

In our implementation, we precomputed the minimum distance to g_{plan} for all vertices of G_{plan} and adopted it as the heuristic function h . Our implementation was conducted in C++, and we used the priority queue and map data structures from the standard library to manage *Open*, *Closed*, and *D*. Dynnikov coordinates were implemented by using the GMP library.

We used Intel(R) Xeon(R) Gold 6338 CPUs @ 2.00 GHz for these experiments.

6.2 Evaluation of Runtime

We measured runtime of our method (**Dyn**) with the following two settings:

- (A) To observe the increase in runtime with the number of agents, we ran our algorithm with $n = 500$ and $K = 100$, where n is the number of agents and K is the number of solutions to find, measuring the runtime from the beginning after each agent's planning process was completed (line 33 in Algorithm 1).
- (B) To observe the increase in runtime with the value of K , we ran our algorithm with $n = 30$ and $K = 1, 2, 3, 10, 3, 100$.

6.2.1 Problem Instances. We evaluated the runtime performance using random instances of the multi-agent pathfinding problem. We used the three grid maps, empty-48-48, den312d, and random-64-64-10, illustrated in Figure 15, from Moving AI MAPF Benchmarks (Stern et al. 2019) as environments. The sizes of the maps were 48×48 , 81×65 , and 64×64 , respectively. The numbers of the connected components of obstacles (r in the previous section) were 0, 4, and 241, respectively. For each of the experiments (A) and (B), we generated 10 instances on each map. To ensure at least some solutions can be found, we imposed a condition on problem instances that guaranteed that the classical RPP would find a solution, i.e., for any $1 \leq i < n$, the agent i can reach the goal without visiting s_{i+1}, \dots, s_n or g_1, \dots, g_{i-1} .

6.2.2 Baseline. We also recorded the maximum absolute value of the Dynnikov coordinates for the first generated solution at the same time. For comparison, we also ran a method (**HR**) that is the same with ours except that it uses the Dehornoy order (Dehornoy 1994) to maintain braids and the handle-reduction algorithm (Dehornoy 1997) to determine the order. This algorithm has long been known as a practically efficient method to solve the word problem for braids, while its theoretical complexity remains unknown (Dehornoy 2008; Dehornoy et al. 2008). More precisely, we used the "FullHRed" algorithm as a specific strategy for the handle reduction method.

6.2.3 Results and Discussion. Figure 17 illustrates the results of the runtimes in the experiment (A), which show that the runtime order of with respect to the number n of agents was smaller for our method than for

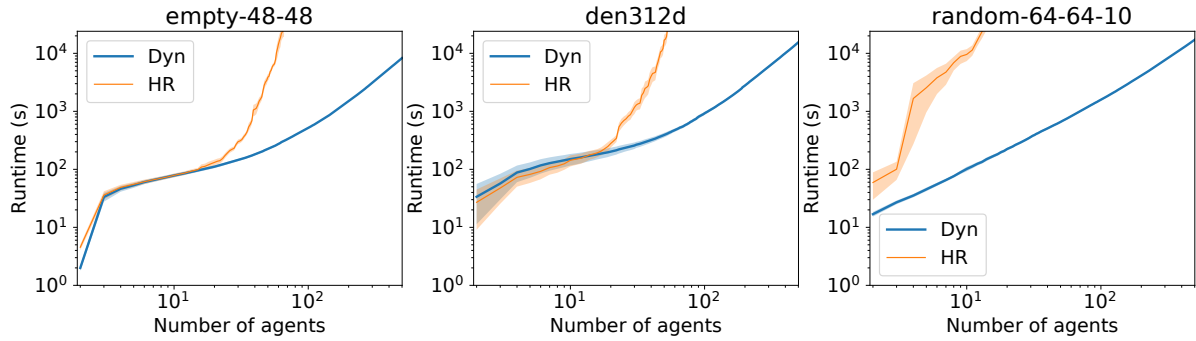


Fig. 17. Log-log plots of runtime with respect to the number of agents for our method and the baseline method using the Dehornoy order. Colored areas represent standard errors.

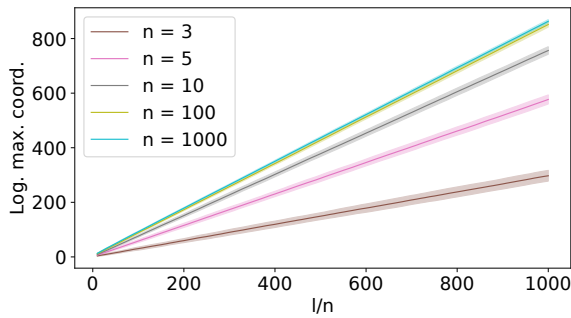


Fig. 18. Logarithms of maximum absolute values of coordinates for random braids with respect to lengths of braid words divided by the number n of agents, for various values of n . Colored areas represent standard deviations.

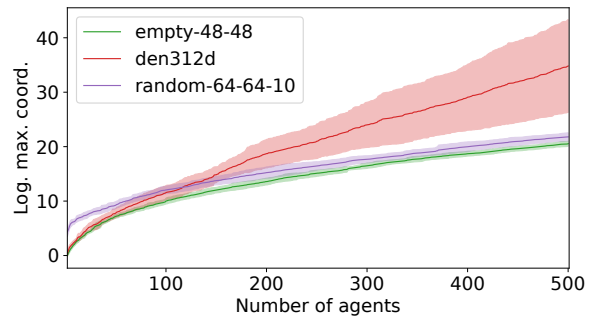


Fig. 19. Logarithms of maximum absolute values of coordinates in experiments with respect to the number of agents, for each environment map. Colored areas represent standard deviations.

HR. Moreover, while the runtime increased significantly in the case with many obstacles (`random-64-64-10`) compared with the other cases in **HR**, it did not change much in our method.

In the case of the empty map, the runtime of **HR** was approximately $\Theta(n^5)$, which means that the runtime for the planning for the n -th agent was $\Theta(n^4)$. This is expected when the handle reduction is the bottleneck because the lengths l of braid words are expected to be roughly $\Theta(n^2)$ and the time complexity of the handle reduction method is conjectured to be $\Theta(l^2)$ (Dehornoy et al. 2008).

On the other hand, the runtime order of our method seemed to be roughly $\Theta(n^2)$, which was smaller than the expected order explained below for calculating Dynnikov coordinates, which means that this calculation was not a bottleneck at the scale of our experiment. Indeed, when we profiled the performance of a single run by using `gprof`, comparison and update of Dynnikov coordinates accounted for 7.2% and 12.3% of runtime, respectively, even in `random-64-64-10`.

As mentioned before, the complexity of one update calculation (11) was $\Theta(\log X)$, where X is the absolute value of the coordinates. While $\log X$ is bounded as $O(l)$, where l is a length of the braid word, for arbitrary words, it is expected to be $\Theta(l/n)$, where n is the number of strands, for random words. To confirm this, we generated 100 random braid words with a length of $1000n$ for each value of $n = 3, 5, 10, 100, 1000$ and calculated

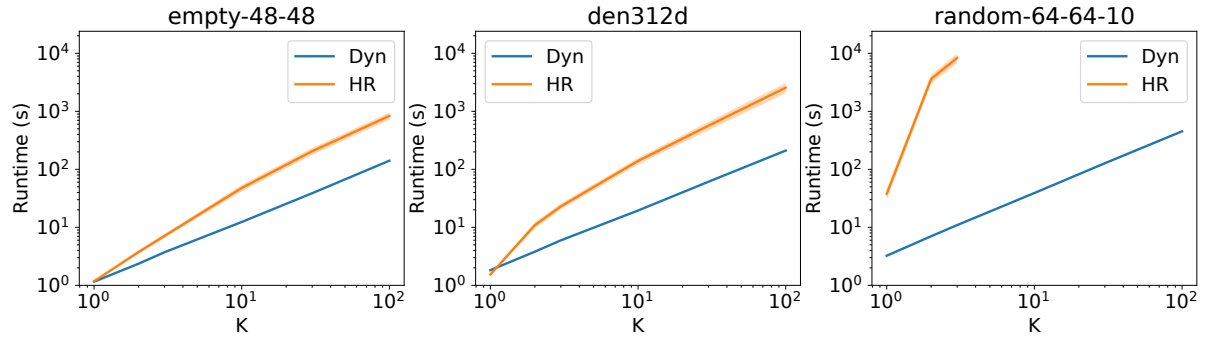


Fig. 20. Log-log plots of runtime with respect to the number K of solutions to find for our method and the baseline method using the Dehornoy order. Colored areas represent standard errors.

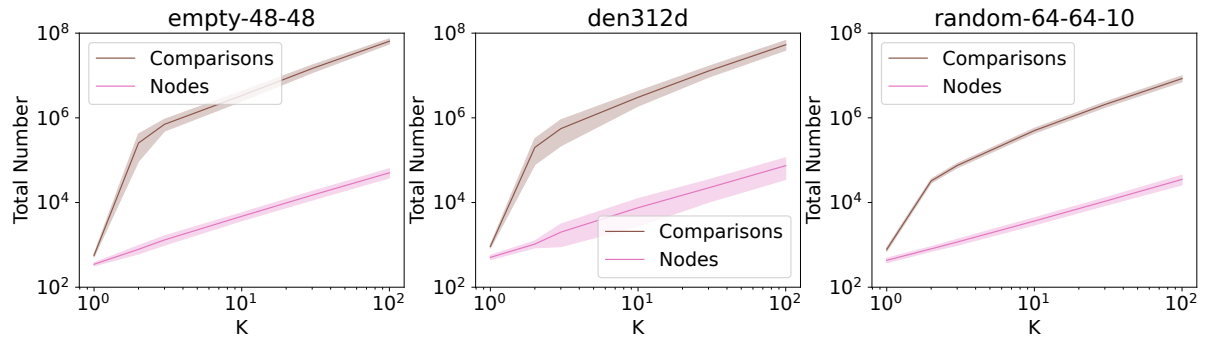


Fig. 21. Log-log plots of the total numbers of braid comparisons and created nodes with respect to the number K of solutions to find. Colored areas represent standard deviations.

the maximum absolute values of the Dynnikov coordinates. The results shown in Figure 18 demonstrate that the expected value of $\log X$ is linear to l/n and that the coefficient is independent of n when n is large enough.

In our method, since the expected lengths of braid words for the planning of the n -th agent is roughly $\Theta(n^2)$, $\log X$ is expected to be $\Theta(n)$, which is consistent with the plot of the logarithms of the maximum absolute values of coordinates with respect to n as shown in Figure 19 when n is enough large. On the other hand, this figure also shows that the coefficients vary depending on the shapes of maps and are larger for complex maps such as den312d. The number of update calculations for the planning for the n -th agent is roughly estimated to be $\Theta(n^2)$, so the runtime for the planning for the n -th agent is expected to be $\Theta(n^3)$. Thus, the total runtime for calculating Dynnikov coordinates for the planning of n agents is expected to be $\Theta(n^4)$. When the number of agents is much larger, this cost will dominate.

Figure 20 presents the runtime results from experiment (B). The runtime in **Dyn** increased approximately linearly with respect to K , whereas in **HR** the increase was slightly steeper than linear. The difference in runtime slope appears to stem from the fact that braid comparisons constituted the bottleneck in **HR**, but not in **Dyn**. Figure 21 illustrates the number of braid comparisons performed and nodes created during algorithm execution¹⁰. Indeed, the number of nodes grew roughly linearly, while the number of comparisons exhibited a slope somewhat

¹⁰Strictly speaking, this is the result in **Dyn**, so the number of comparisons is not exactly the same in **HR**, but there is no essential difference.

steeper than linear. Since braid comparisons are performed only for nodes on the same timed graph G_{plan} , and the number of such timed graphs is K , one would expect the number of comparisons to grow linearly with K if the searches were distributed evenly across the graphs. The deviation from linearity therefore suggests that the exploration was uneven.

The results of experiments (A) and (B) confirmed that our method **Dyn** is clearly superior to the baseline method **HR** in scalability with respect to the number of agents, and also outperforms it in scalability with respect to the number of solutions K to be found.

6.3 Optimization Experiment

To confirm that generating multiple homotopically distinct solutions is useful for solving practical problems, we conducted an experiment in this subsection.

We considered the multi-agent trajectory optimization problem in a continuous planar domain and used the following strategy to solve it.

- (1) We generate multiple solutions on grids.
- (2) We continuously optimize the found solutions and select the best optimized trajectories.

To generate initial solutions, we ran our algorithm to produce homotopically distinct solutions, and compared them with baseline methods.

6.3.1 Problem Setting. We consider multi-agent trajectory optimization in a continuous planar domain, where the objective function is based on acceleration (Zucker et al. 2013). This objective function can be justified from a practical point of view because the energy required for an agent to move and the force felt by something inside the agent depend on its acceleration.

Let n be the number of agents. Let $\mathcal{D} \subseteq \mathbb{R}^2$ be the domain in which the agents can move. As in § 4, we assume that \mathcal{D} has the form $D_0 \setminus (O_1 \cup \dots \cup O_r)$. Let r be the collision radius of the agents. Let d_l be the distance function from O_l for $l > 0$ and that from the outside boundary for $l = 0$. The start positions $s_1, \dots, s_n \in \mathcal{D}$ and goal positions $g_1, \dots, g_n \in \mathcal{D}$ for all agents are given.

The goal is to find collision-free trajectories for all agents with the lowest sum of costs. The cost is defined as the integral of the squared acceleration norm over time, which is regularized to the unit interval. In summary, we want to find $\gamma_1, \dots, \gamma_n : [0, 1] \rightarrow \mathcal{D}$ that minimize the following cost

$$\frac{1}{2} \sum_{i=1}^n \int_0^1 \left\| \frac{d^2 \gamma_i}{dt^2}(t) \right\|^2 dt \quad (16)$$

under the following conditions:

$$\gamma_i(0) = s_i, \gamma_i(1) = t_i, \frac{d\gamma_i}{dt}(0) = \frac{d\gamma_i}{dt}(1) = \mathbf{0}, \quad (17)$$

$$\|\gamma_i(t) - \gamma_j(t)\| \geq 2r \text{ when } i \neq j, \quad (18)$$

$$d_l(\gamma_i(t)) \geq r. \quad (19)$$

6.3.2 Problem Instances. We used two small grid maps: an empty 14×14 map and a handmade map of the same size with five obstacles as shown in Figure 16. We set $r = \sqrt{2}/4$, which is the maximum radius at which no collisions occur in grid map solutions. For each map, we generated 100 problem instances with start positions selected at random to be distinct and similarly selected goal positions.

6.3.3 Compared Methods. To solve the above problem instances, we first generated initial plans and then optimized them in the way described in the next subsection. As a method to generate initial plans, we compared the following three approaches:

- **Ours:** For each instance, we generated 100 homotopically distinct solutions with our method ($K = 100$).
- **Optimal one (OO):** We generated one optimal solution on the grid for each instance.
- **Revised prioritized planning with various priority orders (PPvP):** We generated 100 solutions for each instance by revised prioritized planning with randomly selected priority orders.

The intention of comparing our strategy to **OO** was to confirm that generating multiple initial plans is effective. Also, we added **PPvP** to the comparison to answer the question of whether it is possible to generate diversified plans by simply generating solutions at random without homotopical consideration.

6.3.4 Optimization Method. After generating the initial solutions, we proceeded to optimize them. To do this, we relax the collision-free conditions (18) and (19) and incorporate them into the cost function (Kasaura, Yonetani, et al. 2023). Let $\gamma_1, \dots, \gamma_n$ be the trajectories of the agents. The function C to be minimized is defined as

$$C(\gamma_1, \dots, \gamma_n) := \frac{1}{2} \sum_{i=1}^n \int_0^1 \left\| \frac{d^2 \gamma_i}{dt^2}(t) \right\|^2 dt + c_c \sum_{i \neq j} \int_0^1 \max \left\{ \frac{1}{\|\gamma_i(t) - \gamma_j(t)\|} - \frac{1}{2r}, 0 \right\}^2 dt + c_c \sum_{i=1}^n \sum_{l=0}^r \int_0^1 \max \left\{ \frac{1}{d_l(\gamma_i(t))} - \frac{1}{r}, 0 \right\}^2 dt, \quad (20)$$

where c_c is a penalty coefficient.

More precisely, we approximate the trajectories of the i -th agent by $(L + 1)$ timed waypoints $(p_{i,0}, 0), (p_{i,1}, 1/L), \dots, (p_{i,L}, 1)$ with $p_{i,0} = s_i$ and $p_{i,L} = g_i$, where $L = 100$. The cost function is then reformulated as

$$C(p_{1,0}, \dots, p_{1,L}, p_{2,0}, \dots, p_{n,L}) := \frac{L^3}{2} \sum_i \sum_{k=0}^L \|p_{i,k+1} + p_{i,k-1} - 2p_{i,k}\|^2 + \frac{c_c}{L} \sum_{i \neq j} \sum_{k=0}^L \max \left\{ \frac{1}{\|p_{i,k} - p_{j,k}\|} - \frac{1}{2r}, 0 \right\}^2 + \frac{c_c}{L} \sum_{i=1}^n \sum_{l=0}^r \sum_{k=0}^L \max \left\{ \frac{1}{d_l(p_{i,k})} - \frac{1}{r}, 0 \right\}^2, \quad (21)$$

where $p_{i,-1} := p_{i,0}$ and $p_{i,L+1} := p_{i,L}$. We solved this continuous optimization problem with the Levenberg-Marquardt (Levenberg 1944; Marquardt 1963) algorithm implemented in `g2o 1.0.0` (Kümmerle et al. 2011). The $p_{i,k}$ values were initialized by the initial plan on the grid. We optimized them with 10000 steps of the Levenberg-Marquardt algorithm. The collision-penalty constant c_c was initially set to 10^6 and multiplied by 1.001 after every optimization step.

6.3.5 Results and Discussion. In Figure 22, for each value of $N = 1, \dots, 100$ on the x -axis, the curve labeled **Ours** and the curve labeled **PPvP** represent the cost of the best plan obtained from optimizing the first N initial plans, averaged over 100 instances. The line labeled **OO** represents the cost of optimization of the optimal plan on the grid, also averaged over 100 instances. Figure 23 shows the winning rates. For each method, the curve represents the percentage of instances where the best plan among the first N plans of all methods is generated by it, where $N = 1, \dots, 100$. If costs are the same, the instance is counted for both methods.

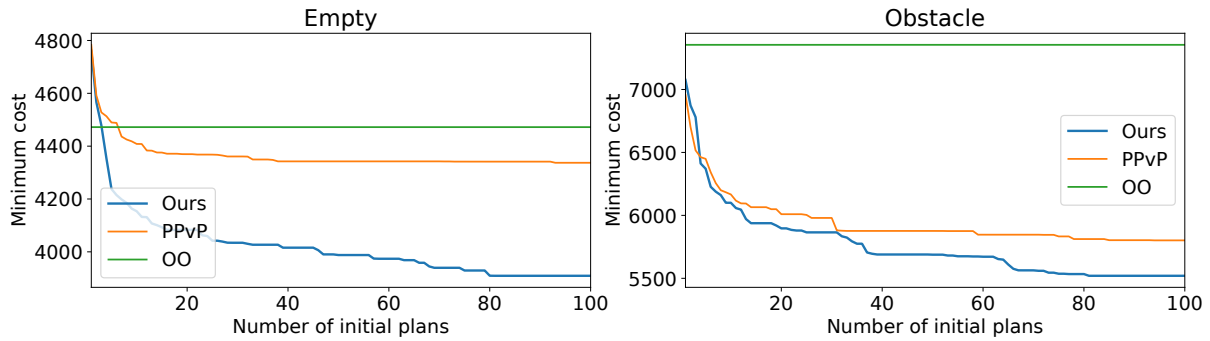


Fig. 22. For **Ours** and **PPvP**, curves represent minimum cost of plans after optimization among first N generated plans, averaged over 100 instances, with respect to N . Bar for **OO** represents the cost after optimization, averaged over 100 instances.

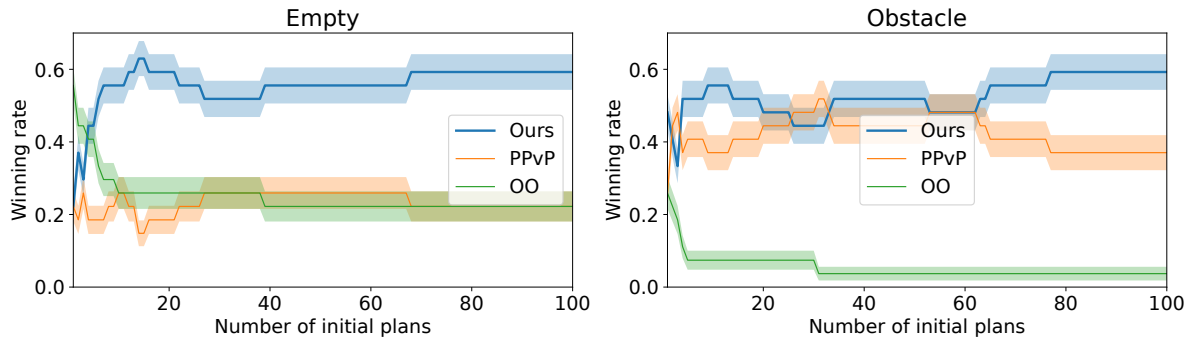


Fig. 23. For each method, curves show rate of instances in which the best of the first N plans generated by it wins over that of other methods, with respect to N . Colored areas represent standard errors.

In the empty map, **OO** tended to produce the best trajectories when only a few initial solutions were considered. Notably, **OO** was not the best even at $N = 1$ in the map with obstacles. This may be because reducing costs by making clever moves on narrow parts of the grid would make the optimized trajectories worse.

For both maps, **Ours** tended to produce better solutions than **PPvP** in terms of both average cost and winning rate. The difference was larger for the empty map than for the map with obstacles.

To confirm that homotopical diversity of initial plans contributed to the minimum cost, we counted the number of homotopically distinct solutions generated with **PPvP** for each instance. The distribution is shown in Figure 24 as histograms. The results indicate that **PPvP** generated few homotopies, particularly on the empty map. This is expected given the lack of homotopy divergence due to obstacles on the empty map. The difference in the homotopical diversity of the plans generated by **PPvP** between maps is thought to be responsible for the difference in results after optimization. Thus, homotopical diversity is actually important for avoiding local optima and obtaining global optima. Our method is particularly effective on maps with few obstacles, where it is difficult to generate homotopically distinct plans naively.

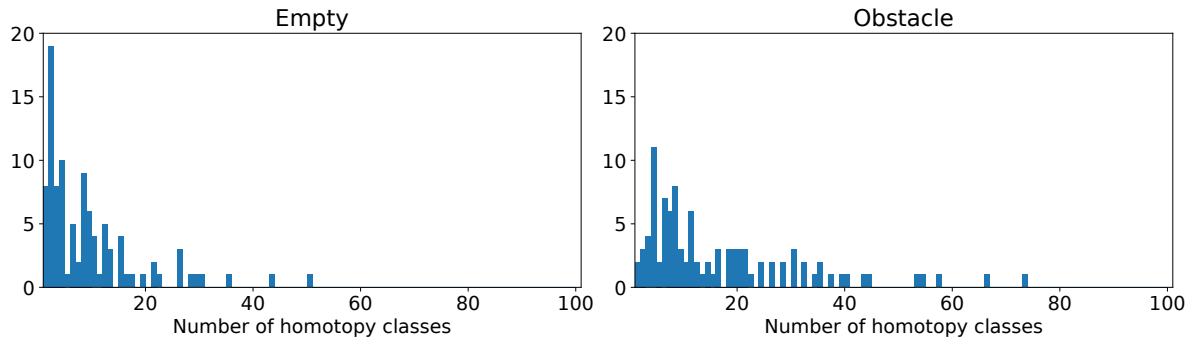


Fig. 24. Histogram of the number of homotopy classes present in 100 solutions generated with **PPvP**, for 100 instances.

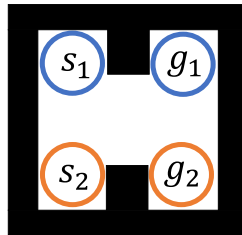


Fig. 25. Example scenarios in which homotopy depends on order in which agents pass through the narrow part.

7 Conclusion and Future Work

We proposed a practical method for multi-agent path planning in planar domains with obstacles. We used Dynnikov coordinates and revised prioritized planning to efficiently generate homotopically distinct plans. Our experimental results indicate that our method is significantly faster than the method using the Dehornoy order. We also conducted an experiment to optimize the generated plans and compared them with those generated with baseline methods. The results indicated that generating homotopically distinct plans using our method led to an improvement in optimized plans. This showcases the effectiveness of our approach in achieving lower-cost trajectories for multi-agent path planning.

One of the main limitations of our approach is that it cannot consider the homotopical differences that arise only when both sizes of agents and obstacles are taken into account. In scenarios where several agents need to pass through a narrow part where only one agent can pass at a time, homotopy classes of solutions can split, depending on the order in which the agents pass through the narrow section. Figure 25 illustrates such a situation, where the homotopy classes of solutions depend on whether the first agent goes through the narrow section earlier or the second agent does. This phenomenon does not occur when the size of agents can be ignored, as assumed with our approach. There is a computational-geometric difficulty when computing these differences.

Another limitation is the abandonment of optimality. While simple A^* search is workable, it becomes impractical for larger problem instances (Stern 2019). To improve efficiency without sacrificing optimality, it may be possible to combine our framework with efficient optimal approaches. To calculate braids while searching, it is required to determine the moves of all agents. This makes it challenging to use conflict-based search, where agent paths are planned independently in its low-level search. On the other hand, increasing-cost tree search could be a more

suitable candidate since it takes into account combinations of all agent paths in its low-level search. Another non-trivial challenge is to make reduction-based methods homotopy-aware.

A potential extension of our research involves exploring decentralized control strategies. Specifically, it may be possible to achieve effective coordination among agents with minimal communication by conveying only braids to other agents.

Acknowledgments

We thank Daniel Harabor and anonymous reviewers for their valuable comments and suggestions, which helped improve the quality of this paper.

References

- A. Adler, M. De Berg, D. Halperin, and K. Solovey. 2015. “Efficient multi-robot motion planning for unlabeled discs in simple polygons.” In: *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*. Springer, 1–17.
- A. Andreychuk, K. Yakovlev, P. Surynek, D. Atzmon, and R. Stern. 2022. “Multi-agent pathfinding with continuous time.” *Artificial Intelligence*, 103662.
- F. F. Arias, B. Ichter, A. Faust, and N. M. Amato. 2021. “Avoidance critical probabilistic roadmaps for motion planning in dynamic environments.” In: *IEEE International Conference on Robotics and Automation*, 10264–10270.
- E. Artin. 1947a. “Braids and permutations.” *Annals of Mathematics*, 643–649.
- E. Artin. 1947b. “Theory of braids.” *Annals of Mathematics*, 101–126.
- R. Barták, N.-F. Zhou, R. Stern, E. Boyarski, and P. Surynek. 2017. “Modeling and solving the multi-agent pathfinding problem in picat.” In: *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 959–966.
- S. Bespamyatnikh. 2003. “Computing homotopic shortest paths in the plane.” *Journal of Algorithms*, 49, 2, 284–303.
- S. Bhattacharya. 2010. “Search-based path planning with homotopy class constraints.” In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 24, 1230–1237.
- S. Bhattacharya and R. Ghrist. 2018. “Path homotopy invariants and their application to optimal trajectory planning.” *Annals of Mathematics and Artificial Intelligence*, 84, 139–160.
- S. Bhattacharya, S. Kim, H. Heidarsson, G. S. Sukhatme, and V. Kumar. 2015. “A topological approach to using cables to separate and manipulate sets of objects.” *The International Journal of Robotics Research*, 34, 6, 799–815.
- S. Bhattacharya, M. Likhachev, and V. Kumar. 2011. “Identification and representation of homotopy classes of trajectories for search-based path planning in 3d.” *Proc. Robot. Sci. Syst.*, 9–16.
- S. Bhattacharya, M. Likhachev, and V. Kumar. 2012. “Topological constraints in search-based robot path planning.” *Autonomous Robots*, 33, 273–290.
- S. Bhattacharya, D. Lipsky, R. Ghrist, and V. Kumar. 2013. “Invariants for homology classes with application to optimal search and planning problem in robotics.” *Annals of Mathematics and Artificial Intelligence*, 67, 3-4, 251–281.
- S. Bigelow. 2001. “Braid groups are linear.” *Journal of the American Mathematical Society*, 14, 2, 471–486.
- J. Birman, K. H. Ko, and S. J. Lee. 1998. “A new approach to the word and conjugacy problems in the braid groups.” *Advances in Mathematics*, 139, 2, 322–353.
- C. Cao, P. Trautman, and S. Iba. 2019. “Dynamic channel: A planning framework for crowd navigation.” In: *2019 international conference on robotics and automation (ICRA)*. IEEE, 5551–5557.
- M. Čáp, J. Gregoire, and E. Frazzoli. 2016. “Provably safe and deadlock-free execution of multi-robot plans under delaying disturbances.” In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5113–5118.
- M. Čáp, P. Novák, A. Kleiner, and M. Selecký. 2015. “Prioritized planning algorithms for trajectory coordination of multiple mobile robots.” *IEEE transactions on automation science and engineering*, 12, 3, 835–849.
- O. De Groot, L. Ferranti, D. M. Gavrila, and J. Alonso-Mora. 2024. “Topology-driven parallel trajectory optimization in dynamic environments.” *IEEE Transactions on Robotics*.
- P. Dehornoy. 1997. “A fast method for comparing braids.” *Advances in Mathematics*, 125, 2, 200–235.
- P. Dehornoy. 1994. “Braid groups and left distributive operations.” *Transactions of the American Mathematical Society*, 345, 1, 115–150.
- P. Dehornoy. 2008. “Efficient solutions to the braid isotopy problem.” *Discrete Applied Mathematics*, 156, 16, 3091–3112.
- P. Dehornoy, I. Dynnikov, D. Rolfsen, and B. Wiest. 2008. “Ordering Braids.” *Mathematical Surveys and Monographs*, 148.
- Y. Diaz-Mercado and M. Egerstedt. 2017. “Multirobot mixing via braid groups.” *IEEE Transactions on Robotics*, 33, 6, 1375–1385.

- K. Dresner and P. Stone. 2008. "A multiagent approach to autonomous intersection management." *Journal of Artificial Intelligence Research*, 31, 591–656.
- I. A. Dynnikov. 2002. "On a Yang-Baxter map and the Dehornoy ordering." *Russian Mathematical Surveys*, 57, 3, 592.
- A. Efrat, S. G. Kobourov, and A. Lubiw. 2006. "Computing homotopic shortest paths efficiently." *Computational Geometry*, 35, 3, 162–172.
- D. B. Epstein. 1992. *Word processing in groups*. CRC Press.
- M. Erdmann and T. Lozano-Perez. 1987. "On multiple moving objects." *Algorithmica*, 2, 1, 477–521.
- E. Fadell and L. Neuwirth. 1962. "Configuration spaces." *Mathematica Scandinavica*, 10, 111–118.
- R. Fox and L. Neuwirth. 1962. "The braid groups." *Mathematica Scandinavica*, 10, 119–126.
- D. Garber, S. Kaplan, and M. Teicher. 2002. "A new algorithm for solving the word problem in braid groups." *Advances in Mathematics*, 167, 1, 142–159.
- F. A. Garside. 1969. "The braid group and other groups." *The Quarterly Journal of Mathematics*, 20, 1, 235–254.
- R. Ghrist. 1999. "Configuration spaces and braid groups on graphs in robotics." *arXiv preprint math/9905023*.
- V. Govindarajan, S. Bhattacharya, and V. Kumar. 2016. "Human-robot collaborative topological exploration for search and rescue applications." In: *Distributed Autonomous Robotic Systems: The 12th International Symposium*. Springer, 17–32.
- J. Gregoire, S. Bonnabel, and A. de La Fortelle. 2013. "Robust multirobot coordination using priority encoded homotopic constraints." *arXiv preprint arXiv:1306.0785*.
- D. Grigoriev and A. Slissenko. 1997. "Computing minimum-link path in a homotopy class amidst semi-algebraic obstacles in the plane." In: *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes: 12th International Symposium, AAECC-12 Toulouse, France, June 23–27, 1997 Proceedings 12*. Springer, 114–129.
- D. Grigoriev and A. Slissenko. 1998. "Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane." In: *Proceedings of the 1998 international symposium on Symbolic and algebraic computation*, 17–24.
- T. Hall and S. Ö. Yurttaş. 2009. "On the topological entropy of families of braids." *Topology and its Applications*, 156, 8, 1554–1564.
- H. Hamidi-Tehrani. 2000. "On complexity of the word problem in braid groups and mapping class groups." *Topology and its Applications*, 105, 3, 237–259.
- A. Hatcher. 2002. *Algebraic Topology*. Cambridge University Press.
- C. Henkel and M. Toussaint. 2020. "Optimized directed roadmap graph for multi-agent path finding using stochastic gradient descent." In: *Annual ACM Symposium on Applied Computing*, 776–783.
- E. Hernandez, M. Carreras, and P. Ridao. 2015. "A comparison of homotopic path planning algorithms for robotic applications." *Robotics and Autonomous Systems*, 64, 44–58.
- W. Hönig, J. A. Preiss, T. S. Kumar, G. S. Sukhatme, and N. Ayanian. 2018. "Trajectory planning for quadrotor swarms." *IEEE Transactions on Robotics*, 34, 4, 856–869.
- L. Jaillet and T. Siméon. 2008. "Path deformation roadmaps: Compact graphs with useful cycles for motion planning." *The International Journal of Robotics Research*, 27, 11–12, 1175–1188.
- K. D. Jenkins. 1991. *The shortest path problem in the plane with obstacles: A graph modeling approach to producing finite search lists of homotopy classes*. Tech. rep. NAVAL POSTGRADUATE SCHOOL MONTEREY CA.
- K. Kasaura, M. Nishimura, and R. Yonetani. 2022. "Prioritized safe interval path planning for multi-agent pathfinding with continuous time on 2D roadmaps." *IEEE Robotics and Automation Letters*, 7, 4, 10494–10501.
- K. Kasaura, R. Yonetani, and M. Nishimura. 2023. "Periodic multi-agent path planning." In: *Proceedings of the AAAI Conference on Artificial Intelligence* 5. Vol. 37, 6183–6191.
- M. Khayyat, A. Zanardi, S. Arrigoni, and F. Braghin. 2024. "Tactical Game-theoretic Decision-making with Homotopy Class Constraints." *arXiv preprint arXiv:2406.13656*.
- S. Kim, S. Bhattacharya, and V. Kumar. 2014. "Path planning for a tethered mobile robot." In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1132–1139.
- S. Kloder and S. Hutchinson. 2006. "Path planning for permutation-invariant multirobot formations." *IEEE Transactions on Robotics*, 22, 4, 650–665.
- B. Knudsen. 2018. "Configuration spaces in algebraic topology." *arXiv preprint arXiv:1803.11165*.
- D. E. Knuth. 1998. *The art of computer programming: Volume 3: Sorting and Searching*. Addison-Wesley Professional.
- D. Krammer. 2002. "Braid groups are linear." *Annals of Mathematics*, 131–156.
- M. Kuderer, C. Sprunk, H. Kretschmar, and W. Burgard. 2014. "Online generation of homotopically distinct navigation paths." In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 6462–6467.
- R. Kümmeler, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. 2011. "g 2 o: A general framework for graph optimization." In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, 3607–3613.
- V. Kurlin. 2012. "Computing braid groups of graphs with applications to robot motion planning." *Homology, Homotopy and Applications*, 14, 1, 159–180.
- E. Lejeune, S. Sarkar, and L. Jezequel. 2021. *A survey of the multi-agent pathfinding problem*. Tech. rep. Technical Report, Accessed July.

- K. Levenberg. 1944. "A method for the solution of certain non-linear problems in least squares." *Quarterly of Applied Mathematics*, 2, 2, 164–168.
- J. Li, A. Tinka, S. Kiesel, J. W. Durham, T. S. Kumar, and S. Koenig. 2020. "Lifelong Multi-Agent Path Finding in Large-Scale Warehouses." In: *International Conference on Autonomous Agents and Multiagent Systems*, 1898–1900.
- J. Lin and J. McCann. 2021. "An artin braid group representation of knitting machine state with applications to validation and optimization of fabrication plans." In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1147–1153.
- J. Liu, M. Fu, A. Liu, W. Zhang, B. Chen, R. Prakapovich, and U. Sychou. 2023. "Homotopy Path Class Encoder Based on Convex Dissection Topology." *arXiv preprint arXiv:2302.13026*.
- R. C. Lyndon, P. E. Schupp, R. Lyndon, and P. Schupp. 1977. *Combinatorial group theory*. Vol. 188. Springer.
- A. Maljutin. 2004. "Fast Algorithms for Identification and Comparison of Braids." *Journal of Mathematical Sciences*, 119, 101–111.
- D. W. Marquardt. 1963. "An algorithm for least-squares estimation of nonlinear parameters." *Journal of the society for Industrial and Applied Mathematics*, 11, 2, 431–441.
- D. Martinez-Baselga, O. de Groot, L. Knödler, L. Riazuelo, J. Alonso-Mora, and L. Montano. 2024. "SHINE: Social homology identification for navigation in crowded environments." *The International Journal of Robotics Research*, 02783649251344639.
- C. Mavrogiannis, J. A. DeCastro, and S. Srinivasa. 2022. "Implicit multiagent coordination at uncontrolled intersections via topological braids." In: *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 368–384.
- C. Mavrogiannis and R. A. Knepper. 2021. "Hamiltonian coordination primitives for decentralized multiagent navigation." *The International Journal of Robotics Research*, 40, 10-11, 1234–1254.
- C. I. Mavrogiannis and R. A. Knepper. 2019. "Multi-agent path topology in support of socially competent navigation planning." *The International Journal of Robotics Research*, 38, 2-3, 338–356.
- C. I. Mavrogiannis and R. A. Knepper. 2020. "Multi-agent trajectory prediction and generation with topological invariants enforced by hamiltonian dynamics." In: *Algorithmic Foundations of Robotics XIII: Proceedings of the 13th Workshop on the Algorithmic Foundations of Robotics 13*. Springer, 744–761.
- P. S. Novikov. 1955. "On the algorithmic unsolvability of the word problem in group theory." *Trudy Matematicheskogo Instituta imeni VA Steklova*, 44, 3–143.
- K. Okumura, R. Yonetani, M. Nishimura, and A. Kanazaki. 2022. "CTRM: Learning to Construct Cooperative Timed Roadmaps for Multi-agent Path Planning in Continuous Spaces." In: *International Joint Conference on Autonomous Agents and Multiagent Systems*, 972–981.
- J. Park, S. Karumanchi, and K. Iagnemma. 2015. "Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming." *IEEE Transactions on Robotics*, 31, 5, 1101–1115.
- D. Peifer. 1997. "An introduction to combinatorial group theory and the word problem." *Mathematics Magazine*, 70, 1, 3–10.
- M. Phillips and M. Likhachev. 2011. "SIPP: Safe interval path planning for dynamic environments." In: *IEEE International Conference on Robotics and Automation*, 5628–5635.
- D. Rolfsen. 2010. "Tutorial on the braid groups." In: *Braids: Introductory Lectures on Braids, Configurations and Their Applications*. World Scientific, 1–30.
- C. Rösmann, F. Hoffmann, and T. Bertram. 2017a. "Integrated online trajectory planning and optimization in distinctive topologies." *Robotics and Autonomous Systems*, 88, 142–153.
- C. Rösmann, F. Hoffmann, and T. Bertram. 2017b. "Kinodynamic trajectory optimization and control for car-like robots." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5681–5686.
- E. Schmitzberger, J.-L. Bouchet, M. Dufaut, D. Wolf, and R. Husson. 2002. "Capture of homotopy classes with probabilistic road map." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 3. IEEE, 2317–2322.
- G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant. 2015. "Conflict-based search for optimal multi-agent pathfinding." *Artificial Intelligence*, 219, 40–66.
- G. Sharon, R. Stern, M. Goldenberg, and A. Felner. 2013. "The increasing cost tree search for optimal multi-agent pathfinding." *Artificial intelligence*, 195, 470–495.
- D. Silver. 2005. "Cooperative pathfinding." In: *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Vol. 1, 117–122.
- R. Stern. 2019. "Multi-agent path finding—an overview." *Artificial Intelligence: 5th RAAI Summer School, Dolgoprudny, Russia, July 4–7, 2019, Tutorial Lectures*, 96–115.
- R. Stern et al. 2019. "Multi-agent pathfinding: Definitions, variants, and benchmarks." In: *Proceedings of the International Symposium on Combinatorial Search*. Vol. 10, 151–158.
- P. Surynek. 2019. "Multi-agent path finding with continuous time viewed through satisfiability modulo theories (SMT)." *arXiv preprint arXiv:1903.09820*.
- P. Surynek, A. Felner, R. Stern, and E. Boyarski. 2016. "Efficient SAT approach to multi-agent path finding under the sum of costs objective." In: *Proceedings of the twenty-second european conference on artificial intelligence*, 810–818.
- J.-L. Thiffeault. 2022. *Braids and Dynamics*. Vol. 9. Springer Nature.
- C. Thomassen. 1992. "The Jordan-Schönflies theorem and the classification of surfaces." *The American Mathematical Monthly*, 99, 2, 116–130.

- B. Tovar, F. Cohen, and S. M. LaValle. 2010. "Sensor beams, obstacles, and possible paths." In: *Algorithmic Foundation of Robotics VIII: Selected Contributions of the Eight International Workshop on the Algorithmic Foundations of Robotics*. Springer, 317–332.
- P. Vernaza, V. Narayanan, and M. Likhachev. 2012. "Efficiently finding optimal winding-constrained loops in the plane." In: *Proceedings of the International Symposium on Combinatorial Search*. Vol. 3, 200–201.
- T. T. Walker, N. R. Sturtevant, and A. Felner. 2018. "Extended Increasing Cost Tree Search for Non-Unit Cost Domains." In: *International Joint Conference on Artificial Intelligence*, 534–540.
- K. Yakovlev and A. Andreychuk. 2017. "Any-angle pathfinding for multiple agents based on SIPP algorithm." In: *Twenty-Seventh International Conference on Automated Planning and Scheduling*, 586–593.
- T. Yang, L. Huang, Y. Wang, and R. Xiong. 2022. "Efficient search of the k shortest non-homotopic paths by eliminating non-k-optimal topologies." *arXiv preprint arXiv:2207.13604*.
- D. Yi, M. A. Goodrich, and K. D. Seppi. 2016. "Homotopy-aware RRT: Toward human-robot topological path-planning." In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 279–286.
- J. Yu and S. M. LaValle. 2013. "Multi-agent path planning and network flow." In: *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*. Springer, 157–173.
- B. Zhou, F. Gao, J. Pan, and S. Shen. 2020. "Robust real-time uav replanning using guided gradient-based optimization and topological paths." In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1208–1214.
- M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa. 2013. "Chomp: Covariant hamiltonian optimization for motion planning." *The International journal of robotics research*, 32, 9-10, 1164–1193.

A On Presentations of Pure Braid Groups

Bhattacharya and Ghrist (2018) gave a presentation of a homotopy group for path planning with n agents on a plane, which is a pure braid group P_n , with the following generators:

$$\{u_{i,j/\gamma_{i+1},\dots,\gamma_{j-1}} \mid 1 \leq i < j \leq n, \gamma_{i+1}, \dots, \gamma_{j-1} \in \{+, -\}\}. \quad (22)$$

The relations are as follows.

- For $i < j < k$, $\alpha_{i+1}, \dots, \alpha_{j-1} \in \{+, -\}$, and $\beta_{j+1}, \dots, \beta_{k-1} \in \{+, -\}$,

$$\begin{aligned} & u_{i,j/\alpha_{i+1},\dots,\alpha_{j-1}} \cdot u_{i,k/\alpha_{i+1},\dots,\alpha_{j-1},-\beta_{j+1},\dots,\beta_{k-1}} \cdot u_{j,k/\beta_{j+1},\dots,\beta_{k-1}} \\ & \cdot u_{i,j/\alpha_{i+1},\dots,\alpha_{j-1}}^{-1} \cdot u_{i,k/\alpha_{i+1},\dots,\alpha_{j-1},+\beta_{j+1},\dots,\beta_{k-1}}^{-1} \cdot u_{j,k/\beta_{j+1},\dots,\beta_{k-1}}^{-1}, \end{aligned} \quad (23)$$

$$\begin{aligned} & u_{i,j/\alpha_{i+1},\dots,\alpha_{j-1}} \cdot u_{i,k/\alpha_{i+1},\dots,\alpha_{j-1},-\beta_{j+1},\dots,\beta_{k-1}} \\ & \cdot u_{i,j/\alpha_{i+1},\dots,\alpha_{j-1}}^{-1} \cdot u_{i,k/\alpha_{i+1},\dots,\alpha_{j-1},+\beta_{j+1},\dots,\beta_{k-1}}^{-1}, \end{aligned} \quad (24)$$

and

$$\begin{aligned} & u_{i,k/\alpha_{i+1},\dots,\alpha_{j-1},-\beta_{j+1},\dots,\beta_{k-1}} \cdot u_{j,k/\beta_{j+1},\dots,\beta_{k-1}} \\ & \cdot u_{i,k/\alpha_{i+1},\dots,\alpha_{j-1},+\beta_{j+1},\dots,\beta_{k-1}}^{-1} \cdot u_{j,k/\beta_{j+1},\dots,\beta_{k-1}}^{-1}. \end{aligned} \quad (25)$$

- Let i, j, i', j' be distinct indices with $i < j$, $i' < j'$, $i < i'$. Let $\gamma_{i+1}, \dots, \gamma_{j-1} \in \{+, -\}$ and $\gamma'_{i'+1}, \dots, \gamma'_{j'-1} \in \{+, -\}$ be signs. When $i < i' < j < j'$, we assume that $\gamma_{i'} \neq \gamma'_{j'}$ and $(\gamma_k, \gamma'_k) \neq (-\gamma_{i'}, \gamma_{i'})$ for all $i' < k < j$. When $i < i' < j' < j$, we assume that $\gamma_{i'} = \gamma_{j'}$ and $(\gamma_k, \gamma'_k) \neq (-\gamma_{i'}, \gamma_{i'})$ for all $i' < k < j'$. For such tuples,

$$u_{i,j/\gamma_{i+1},\dots,\gamma_{j-1}} \cdot u_{i',j'/\gamma'_{i'+1},\dots,\gamma'_{j'-1}} \cdot u_{i,j/\gamma_{i+1},\dots,\gamma_{j-1}}^{-1} \cdot u_{i',j'/\gamma'_{i'+1},\dots,\gamma'_{j'-1}}^{-1}. \quad (26)$$

The description in the original paper is incomplete as it omits the condition for the relation (26).

We consider the word

$$w = u_{1,4/-} u_{2,4/-} u_{3,4} u_{1,4/+}^{-1} u_{2,4/+}^{-1} u_{3,4}, \quad (27)$$

which is irreducible when using Dehn's algorithm. On the other hand,

$$\begin{aligned}
w &= u_{2,4/-} u_{1,4/+} u_{3,4}^{-1} u_{1,4/++}^{-1} u_{2,4/+}^{-1} u_{3,4}^{-1} \\
&= u_{2,4/-} u_{3,4}^{-1} u_{2,4/+}^{-1} u_{3,4}^{-1} \\
&= u_{3,4}^{-1} u_{3,4}^{-1} \\
&= e,
\end{aligned} \tag{28}$$

where the first, second, and third equalities are deduced from (25) with $(i, j, k) = (1, 2, 4)$, $(1, 3, 4)$, and $(2, 3, 4)$, respectively. Thus, Dehn's algorithm is incomplete for this presentation when $n \geq 4$.

The pure braid group P_n has a standard presentation (Rolfsen 2010) using generators $\{a_{i,j} \mid 1 \leq i < j \leq n\}$ with notation (12) and the following relations.

- For $1 \leq i < j < k \leq n$,

$$a_{i,j} a_{i,k} a_{j,k} a_{i,j}^{-1} a_{j,k}^{-1} a_{i,k}^{-1}, \tag{29}$$

and

$$a_{i,k} a_{j,k} a_{i,j} a_{i,k}^{-1} a_{i,j}^{-1} a_{j,k}^{-1}. \tag{30}$$

- For $1 \leq i < j < k < l \leq n$,

$$a_{i,j} a_{k,l} a_{i,j}^{-1} a_{k,l}^{-1}, \tag{31}$$

$$a_{i,l} a_{j,k} a_{i,l}^{-1} a_{j,k}^{-1}, \tag{32}$$

and

$$a_{i,k} a_{j,k} a_{j,l} a_{j,k}^{-1} a_{i,k}^{-1} a_{j,k} a_{j,l}^{-1} a_{j,k}^{-1}. \tag{33}$$

Let

$$u_{i,j/\sigma_{i+1}, \dots, \sigma_{j-1}} = a_{i,i+1}^{d_{i+1}} \cdots a_{i,i+j-1}^{d_{j-1}} a_{i,j} a_{i,j-1}^{-d_{j-1}} \cdots a_{i,i+1}^{-d_{i+1}}, \tag{34}$$

where $d_k = 1$ if $\sigma_k = +$, and $d_k = 0$ if $\sigma_k = -$. Then, the relations for $\{u_{i,j/\sigma_{i+1}, \dots, \sigma_{j-1}}\}$ can be deduced from the relations for $\{a_{i,j}\}$ and vice versa. Therefore, we can translate words from Bhattacharya and Ghrist's presentation to words for the standard presentation. However, this translation increases the lengths of words by a factor of $O(n)$. On the other hand, lengths of words constructed by Bhattacharya and Ghrist's method are equal to those of words constructed by the method in § 5.3. To the best of our knowledge, there is no algorithm for the word problem of the pure braid group that is efficient enough to compensate for these drawbacks. This is why we use elements of the braid group to label homotopy classes, instead of those of the pure braid group.

B Proof of Proposition 4

The map from $C_{r+n}(\mathbb{R}^2)$ to $C_r(\mathbb{R}^2)$, which sends (p_1, \dots, p_{r+n}) to (p_1, \dots, p_r) , is a fiber bundle (Fadell and Neuwirth 1962), and the fiber at (o_1, \dots, o_r) is the image of the embedding (8). Thus, the following exact sequence of homotopy groups is induced:

$$\pi_2(C_r(\mathbb{R}^2)) \rightarrow \pi_1(C_n(\mathbb{R}^2 \setminus \{o_1, \dots, o_r\})) \rightarrow \pi_1(C_{r+n}(\mathbb{R}^2)), \tag{35}$$

where $\pi_2(C_r(\mathbb{R}^2))$ is the second homotopy group of $C_r(\mathbb{R}^2)$ (Hatcher 2002). Moreover, $\pi_2(C_r(\mathbb{R}^2))$ is trivial (Knudsen 2018).

Received 22 May 2025; accepted 08 January 2026