Summer 2023

Near-Optimal Motion Planning Algorithms Via A Topological and Geometric Perspective

Aakriti Kumari Upadhyay
University at Albany, State University of New York, aupadhyay@albany.edu

NEAR-OPTIMAL MOTION PLANNING ALGORITHMS VIA A TOPOLOGICAL AND GEOMETRIC PERSPECTIVE

by

AAKRITI KUMARI UPADHYAY

A Dissertation

Submitted to the University at Albany, State University of New York

in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Philosophy

College of Engineering and Applied Sciences

Department of Computer Science

Summer 2023

DEDICATED

To God - for giving me this opportunity.

To my parents - for their constant encouragement and support throughout this journey.

To my siblings - for being there to listen to me and for the moral support during tough times.

ABSTRACT

Motion planning is a fundamental problem in robotics, which involves finding a path for an autonomous system, such as a robot, from a given source to a destination while avoiding collisions with obstacles. The properties of the planning space heavily influence the performance of existing motion planning algorithms, which can pose significant challenges in handling complex regions, such as narrow passages or cluttered environments, even for simple objects. The problem of motion planning becomes deterministic if the details of the space are fully known, which is often difficult to achieve in constantly changing environments. Sampling-based algorithms are widely used among motion planning paradigms because they capture the topology of space into a roadmap. These planners have successfully solved high-dimensional planning problems with a probabilistic-complete guarantee, i.e., it guarantees to find a path if one exists as the number of vertices goes to infinity. Despite their progress, these methods have failed to optimize the sub-region information of the environment for reuse by other planners. This results in re-planning overhead at each execution, affecting the performance complexity for computation time and memory space usage.

In this research, we address the problem by focusing on the theoretical foundation of the algorithmic approach that leverages the strengths of sampling-based motion planners and the Topological Data Analysis methods to extract intricate properties of the environment. The work contributes a novel algorithm to overcome the performance shortcomings of existing motion planners by capturing and preserving the essential topological and geometric features to generate a homotopy-equivalent roadmap of the environment. This roadmap provides a mathematically rich representation of the environment, including an approximate measure of the collision-free space. In addition, the roadmap graph vertices sampled close to the obstacles exhibit advantages when navigating through narrow passages and cluttered environments, making obstacle-avoidance path planning significantly more efficient.

The application of the proposed algorithms solves motion planning problems, such as sub-optimal planning, diverse path planning, and fault-tolerant planning, by demonstrating the improvement in computational performance and path quality. Furthermore, we explore the potential of these algorithms in solving computational biology problems, particularly in

finding optimal binding positions for protein-ligand or protein-protein interactions. Overall, our work contributes a new way to classify routes in higher dimensional space and shows promising results for high-dimensional robots, such as *articulated linkage robots*. The findings of this research provide a comprehensive solution to motion planning problems and offer a new perspective on solving computational biology problems.

ACKNOWLEDGMENT

The research work in this dissertation is the result of my consistent hard work and dedication, which would not have been possible without the support of my family, friends, and colleagues. I apologize if I unintentionally left anyone out of the acknowledgments.

I want to express my special appreciation and thanks to my advisor, Dr. Chinwe Ekenna, for her unwavering support and guidance since I joined her for my master's project in January 2017. Under your mentorship, I have gained valuable professional advice and learned to step out of my comfort zone. Your encouragement and direction have led to my numerous successes, including completing my master's research project, becoming a finalist at the Microsoft Student Research Competition (SRC), attending conferences, and my Ph.D. degree.

To my committee members, Dr. Boris Goldfarb, Dr. Mukulika Ghosh, Dr. Paliath Narendran, and Dr. Jeong-Hyon Hwang, thank you for your invaluable contributions during my research development. Your unwavering commitment to challenging my ideas has undoubtedly played a pivotal role in shaping the outcome of this study. I am immensely grateful for the collaborative spirit that has characterized our work together, fostering an environment conducive to intellectual growth and innovation. I also extend my heartfelt thanks for the continuous support you have provided me in various endeavors over the past few years.

I would also like to thank my parents. My father, Mahendra Upadhyay, for being a great source of encouragement. You have always pushed me to think creatively and believe in myself. I am grateful for your guidance and support. Equally deserving of recognition is my mother, Shobha Upadhyay. Your selfless sacrifices, boundless love, and prayers have played an integral role in shaping me into the person I am today. I want to express my heartfelt gratitude to both of you for your unconditional love and support. You never said "no" to my endeavors, and you never told me that I had to be anyone else than who I was. I love you, and I am proud to be your daughter.

To my siblings, Animesh and Iccha, thank you for being a constant source of moral

support and always motivating me. I could not have imagined having a better brother and sister. I love you.

To my supervisors, thank you for your guidance and the chance to gain valuable professional experience. In particular, I would like to acknowledge Savio Joe, Paul Zachos, Kalyan Perumalla, and Joe Morelli for their exceptional supervision. Thanks to your invaluable support, I was able to polish my programming skills and gain experience working with cutting-edge industrial technologies.

To my friends inside and outside the research lab, thank you. Each one of them, past and present, has taught me something valuable and positively impacted my life. I would like to specially name Michael Phipps, Tuan Tran, Sourav Dutta, and Janice M Fontanella for their contributions to my personal and professional growth. Thank you all for being a part of my journey.

Finally, I would also like to thank the chair and administrative staff members at the Department of Computer Science, University at Albany, State University New York, for their invaluable assistance and support.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

In a dynamic environment, autonomous systems face challenges due to constantly changing details. To find a feasible pathway in such a scenario, a vast amount of information needs evaluation which can be tedious when environmental changes are frequent. The performance of these systems also degrades, and computational costs increase with every new occurrence in the environment. Therefore, it becomes essential to focus on gathering crucial information from the underlying space to develop efficient solutions for these systems. Motion planning is a problem of finding a route between two points in the planning space while ensuring its validity. Research has shown that the planning methodologies are PSPACE-complete, implying that planning optimal motions for simple objects, e.g., a car, is computationally challenging. The properties of the environment play a critical role in affecting the performance of these motion planning paradigms. To mitigate this problem, we focus on studying and analyzing the properties of the planning space to extract the intricate details for guided, efficient path planning. We identify and capture the features of the environment by using Topological Data Analysis (TDA) [91] methods and approximate the mathematical representation of the space in a roadmap (graph). In this process, we developed sophisticated solutions to different motion planning problems, such as sub-optimal planning, diverse path planning, and fault-tolerant planning, while also addressing issues from structural biology. A detailed discussion is available from Chapter 3.

Studies show that motion planning can become deterministic if the properties of planning space are fully known. However, examining all possible movements of a robot in a three-dimensional mathematical space is challenging due to the lack of minute descriptions of the position and orientation of the robot's joints, which makes it difficult to verify and validate the motions or faults of high-dimensional robots, such as manipulators or humanoid robots. We address this issue by focusing our research on the configuration space ($\mathcal{C}_{space}$) [82]. This space captures the environment as it is and transforms it into the robot's dimensionality, where the robot itself becomes a point object. As shown in Figure 1.1, the dimensionality of space is the robot's dimensionality, which aids in examining the robot's orientation at each

dimension. By analyzing our planners in the configuration space, we better understand the motion planning of high-dimensional robots and improve their performance for real-world applications.

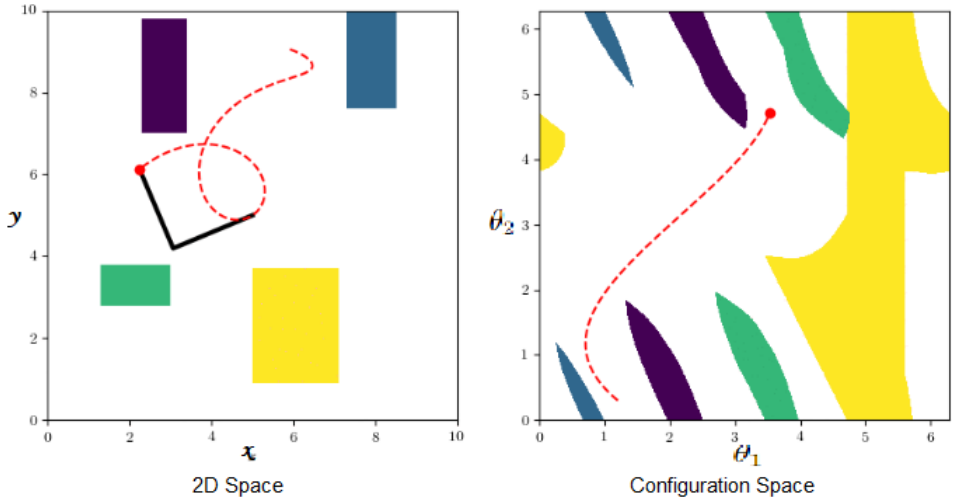


Figure 1.1: On the left is a two-dimensional x-y plane having 2 DOF (degrees of freedom) robot with joint angles, i.e., θ_1 and θ_2. Towards the right is the configuration space on the θ_1 and θ_2 axis with the robot as a point object.

A robot is a movable object whose position and orientation can be described by n parameters, or *degrees of freedom* ($DOFs$), each corresponding to an object component (e.g., object positions, object orientations, joint angles, joint displacements). Hence, a robot's placement, or configuration, can be uniquely described by a point $(x_1, x_2, ..., x_n)$ in an n dimensional space (x_i being the ith DOF). The subset of all feasible (collision-free) configurations is the free space ($\mathcal{C}_{free}$), and the union of the unfeasible configurations (i.e., the robot cannot traverse through these configurations) is the blocked or obstacle space ($\mathcal{C}_{obst}$). A $\mathcal{C}_{space}$ consists of topological features that are defined as the basic representation of a mathematical space and refer to features that support continuity, connectivity, and convergence that are established and maintained based on geometric coincidence. The properties of $\mathcal{C}_{space}$ can be divided into quantitative type, i.e., geometry (relates to the measurements of distances, angles, and areas), and qualitative type, i.e., topology (related to connectivity and compactness).

Topological properties: Topology, sometimes even called "qualitative geometry,"is a branch of mathematics that studies properties of space preserved under a homeomorphism.

In other words, topology is concerned with the qualitative study of how points, sets, and objects are internally and mutually interconnected or how they are adjacent to each other. It is characterized by its abstract nature that is independent of the structure of a metric space, i.e., it is very roughly what is left of geometry when we take away from it everything that has some size. A simple example of a topological graph of a torus-shaped space is shown in Figure 1.2. The graph represents the torus-shaped space by capturing the essential features, i.e., topological properties, and ignoring its extraneous details.

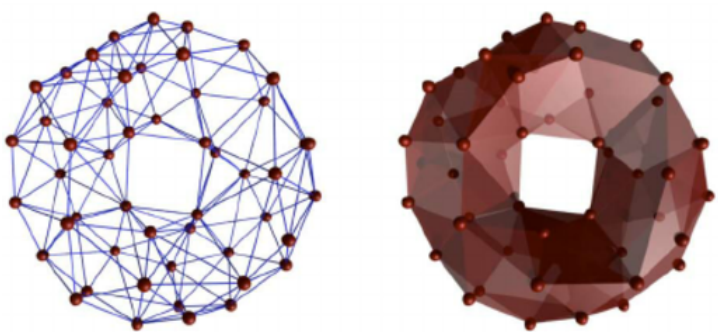


Figure 1.2: The vertices and edges are the graph components that define the topology of a space. The picture on the left shows the topology of the torus-shaped space.

Geometric properties: Geometry deals with the body measurement of an object concerning length, shape, angle, area, volume, and distance. The "scene" in which such measurements are made is space, and we declare some common geometric properties of measured bodies to be geometric properties of this space. The geometric features of a body are the points on its surface that define its shape. Hence, we extract these quantitative features to describe an obstacle's shape in the environment. Figure 1.3 illustrates the local minimum, maximum, and saddle points detected on the object's surface. We refer to them as critical points.

1.1 Motivation

Topology and Geometry have together been the spine behind sophisticated motion planning algorithms. However, it is noteworthy that the investigation of topology and geometry was relatively limited in the past. This lack of attention has hindered the study and comprehension of the limitations of motion planners in dealing with complex regions such

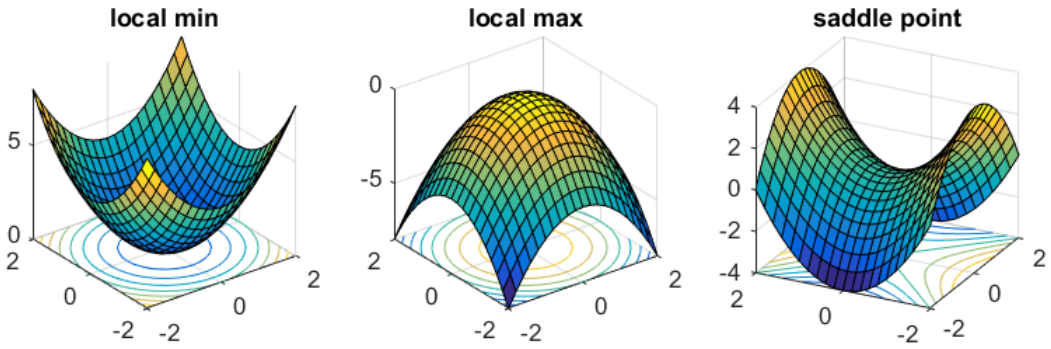


Figure 1.3: From left to right, the geometric features capture depth, peak, and uniform areas of the surface.

as narrow passages or cluttered environments. Additionally, the inability to reuse roadmap information of visited areas of the environment by existing planners has led to a significant computational time and memory usage overhead. Therefore, it is imperative to develop an algorithm that can efficiently extract the properties of $\mathcal{C}_{space}$ while simultaneously enhancing the performance of motion planning methods.

A widely used motion planning method is the sampling-based motion planning approach [39], which attempts to *approximate* the configuration space connectivity information, with geometric properties embedded in the roadmap. The configurations are retained in free space if the connection edge between two samples is collision-free. These algorithms have been successful in high-dimensional environments and are known to be probabilistic complete, i.e., guarantees to find a solution if one exists. However, the roadmap generated by many sampling-based motion planning methods does not describe the configuration space, but only a *subspace* of it. The relation of this subspace to the $\mathcal{C}_{space}$ is primarily unknown, thus making the extracted information of the underlying space of less or no use for future analysis.

In our previous research endeavor [118], we integrated machine learning techniques, specifically reinforcement learning, with sampling-based methods to explore their efficacy in the performance of these planners. The results of our study demonstrated a significant improvement in computation time when utilizing the topological information of the planning space. However, the issue of reusing roadmap information and ensuring traversal-efficient configuration nodes in the roadmap remained unresolved. To mitigate this problem, we leverage the emerging application of TDA tools to enhance the quality of motion planning.

Specifically, we propose a topology-aware sampling-based motion planner that utilizes TDA tools, namely the Vietoris-Rips complex and discrete Morse theory, to extract the topological and geometric properties of the configuration space ($\mathcal{C}_{space}$). The work contributes novel methods and tools that improve the performance of these planners in terms of time and space complexity and provides a comprehensive solution to various motion and path planning problems.

1.2 Research Objective

Our research objective is to comprehend the characteristics of the robot's environment and encapsulate these attributes into a roadmap for memory-efficient path planning. The concept of extracting the properties of the configuration space is not new, and various studies have presented favorable outcomes [36, 153]. The concurrent development of TDA tools has yielded some promising results [13, 3, 4] to address this problem. However, little effort is devoted to using the capabilities of TDA tools to investigate the core issues of motion planning problems that consistently impact the performance of autonomous systems. Specifically, the question of finding the sufficient compact representation of space required to plan a route while disregarding extraneous details persists. By bringing theory into practice, we concentrate on building the theoretical foundation of our algorithm to bridge the gap between mathematical theory-heavy (data topology) research and high application content (robot motion planning). In our initial work, we introduce a novel approach that leverages the strengths of sampling-based methods and TDA tools to effectively approximate the topological and geometric features of the planning space into a roadmap for guided path planning. The generated roadmap has demonstrated significant potential in near-optimal path planning, making it a valuable tool for various applications, including diverse path planning [122], fault-tolerant path planning [121], structural biology [125, 120], and more.

The extension of our framework to an incremental path planner addresses the problem of finding a pathway in a partially observable space. This planner finds usability in real-time situations where the robot needs to plan its path incrementally while moving around in its world. The potential applications are vast, including but not limited to disaster-relief scenarios [26], self-driving cars [15], indoor assistant robots [76], warehouse robots, etc. By incrementally planning the path, it adapts to the changing environment and determines a

feasible and efficient route in its field of view. The planner locally extracts the topological and geometric features of the subspace and metrically glues them together to find a global solution between start and goal positions as the robot moves around the environment. The incremental nature of the planner enables it to find a sub-optimal solution in the planning space without losing critical information.

The emergence of service robots has revolutionized how humans interact with machines. These robots, designed to perform tasks such as vacuuming, valet parking, lawn mowing, and office assistance, represent a new generation of robots that work close to humans. It is a significant departure from the previous industrial robots that were confined to cages and performed repetitive tasks with minimal human interaction. In the current epoch of human-robot interaction, it is of utmost importance to address safety concerns to ensure that robots can operate efficiently in close proximity to humans. So, it is crucial to optimize the clearance between the robot and obstacles in the environment, including humans, while the robot is performing its designated task. The robot must possess the ability to comprehend the features of the environment to finish its work without any hindrance while simultaneously keeping track of safety measures as it moves around. This research develops solutions to such scenarios and provides a sophisticated motion planning paradigm with significant contributions, as mentioned below.

- Compact representation of the space (sample-efficient sparse roadmap) that represents topological (samples and collision-free connections), as well as geometric information (local minima or maxima on obstacle surface).
- Geometric features of the space that governs the quality of the representation, the space approximation, obstacle clearance, and path quality.
- A topology-based path planner that incrementally extracts and glues the topological and geometric information of the partially observable space for memory-efficient path planning.
- A new way to identify and classify coarsely diverse paths in high-dimensional configuration space.
- A fault-tolerant planner to recover an alternate route with minimum re-configurations using the visibility of configuration nodes, the expansiveness of edges, and path length.

- A computational method to identify the geometrically-fitting binding position during bio-molecule interactions.

In summary, this work provides a detailed and precise description of the fundamental properties of the environment or $\mathcal{C}_{space}$. Our findings contribute to the ongoing research in robotics and provide an algorithmic foundation for developing future robotic systems. The theorems, lemmas, and propositions mentioned without citations are the contributions of this research.

1.3 Outline

Chapter 2 discusses the foundational knowledge of sampling-based methods, topological data analysis tools, and current approaches to solving the motion planning problem. Chapter 3 discusses the algorithmic foundation for our methods to extract the topological and geometric properties of configuration space using TDA tools with an extension to incremental planning framework. The applications of our framework to the motion planning problems are presented in Chapter 4, and we discuss extensions of the framework to computational biology in Chapter 5. Finally, we conclude and discuss future work in Chapter 6.

CHAPTER 2

Literature Review

Data extraction tools have played a crucial role in improving path planning in robotics. The research contributions in this area have led to the development of efficient algorithms and tools that can navigate complex environments. In this chapter, we discuss some influential research work in motion planning that leverage the properties of the underlying space to generate efficient routes.

2.1 Sampling-based Motion Planners (SBMP)

Sampling-based methods are broadly classified into two main classes: graph-based methods such as the Probabilistic Roadmap Method (PRM) [68] and tree-based methods such as Rapidly-exploring Random Tree (RRT) [77]. In the following sections, we highlight a few of the PRM and RRT variants most relevant to this research.

2.1.1 Graph-based Methods

PRM constructs a map of $\mathcal{C}_{free}$ by first sampling robot configurations in $\mathcal{C}_{space}$ and connects them to form a graph (roadmap) containing feasible routes. These planners are particularly suited for solving many start and goal queries in the same environment and are easily implementable, computationally efficient, and applicable to a wide range of robots.

Various sampling strategies have been proposed to address the challenge of motion planning in narrow passages. Each method has its advantages and limitations. The choice of sampling method depends on the specific requirements of the application. The uniform sampling method [68] generates nodes uniformly at random in $\mathcal{C}_{space}$ on retaining valid ones. However, it shows an inability to sample in narrow passages efficiently, which leads to the chances of oversampling in other regions. Obstacle-Based PRM (OBPRM) [143] samples configurations near $\mathcal{C}_{obst}$ surfaces by pushing configuration nodes to the $\mathcal{C}_{obst}$ boundary or finding surface intersections of randomly placed line segments. Even if OBPRM excels in

narrow passages, it can be expensive because it requires many validity tests. Gaussian [28] and Bridge Test [64] filter samples with inexpensive tests to find samples near $\mathcal{C}_{obst}$ boundaries or directly in narrow passages, respectively. However, despite their effectiveness, both methods perform similar sampling as uniform random sampling and suffer from needing many samples to find one in a narrow corridor. Additionally, both approaches suffer from parameter tuning, which significantly affects the performance and quality of the mappings produced.

PRM^* [67] is a PRM variant that finds asymptotically optimal paths according to some cost function, i.e., it guarantees to find the shortest pathway if one exists as the number of samples goes to infinity. It differs from PRM in two ways: the number of neighbors considered for connection is a function of the current roadmap size (instead of fixed), and it attempts all possible connections, even if they are already in the same connected component (most PRM implementations ignore such neighbors for efficiency). The cost function is typically path length. Two variants of PRM^*, namely k-PRM^* [67] and LazyPRM^* [61], were designed to enhance the performance of PRM^* by reducing the number of connected neighbors or collision checks. The former variant establishes connections to k-nearest neighbors of a node based on some distance metric, where k is a small constant. On the other hand, the latter variant avoids checking edges that hold no potential to improve the current best path, thereby reducing the per-sample computational cost and accelerating convergence. Although PRM^* produces asymptotically optimal pathways, in practice, it requires a large roadmap to do so and thus is computationally expensive.

Dynamic Region-based PRM [105] combines dynamic region sampling with PRM to connect multiple connected components for effectual solutions with complete coverage. This work extends from a previous work [44] that constructs Delaunay-triangulation to form Reeb graphs for a dynamically growing region. While this method reduces the number of configurations required to compute a solution, it performs computationally-intensive geometric tests to accept samples into the roadmap, resulting in large planning times.

2.1.2 Tree-based Methods

RRT method grows a tree outwards from a root configuration and expands to the unexplored regions of $\mathcal{C}_{space}$ to find a path. These methods are tailored to solve single-query

motion planning problems and have an exponential convergence to the sampling distribution over $\mathcal{C}_{free}$ because of Voronoi bias, i.e., RRTs explore $\mathcal{C}_{free}$ efficiently. Despite these important properties, RRTs suffer in the presence of narrow passages or very complex planning systems.

To solve single query problems in dynamic environments, Dynamic-Domain RRT [147] dynamically grows Voronoi-based regions and biases the random node selection to be within a radius r to enable expansion. The radius is determined dynamically from the failed expansion attempts. Dynamic Region-biased RRT [44] uses the same idea to bias growth using Reeb graphs. The algorithm performs workspace decomposition and presents the $\mathcal{C}_{obst}$ into holes in an embedded roadmap, thus, restricting its usage in non-uniform environments.

RRT^* [67] is an approach to ensure the asymptotic optimality of the tree. RRT^* expands in the same way as RRT except that after expansion, the tree will locally "rewire" itself to optimize the cost function. RRT^*-Connect [69] is a variant of RRT^* that constructs two trees, one rooted at a start configuration and the other at a goal configuration. Each tree grows towards the other using a greedy heuristic. Once the two trees meet, a path gets extracted between the start and the goal using a simple path search algorithm in the tree. This work showed the benefit of allowing RRTs to grow bi-directionally with variable step sizes in their greedy heuristic. Informed RRT^*-Connect [86] operates similarly as RRT^*-Connect except it uses the idea from Informed-RRT [58] to expand trees within the ellipsoidal curve that covers the start and goal configurations. Even though RRT^* performs quite effectively in finding asymptotically shortest paths, it requires many iterations to produce near-optimal solutions.

2.1.3 Hybrid Techniques

Sampling-based Roadmap of Trees (SRT) [99] approaches proposed in the past have successfully combined PRMs and RRTs to achieve scalability on high-performance computers or more effectively explore narrow passages. These approaches provide a powerful tool for motion planning in complex environments. The SRT approach utilizes global sampling techniques, such as PRM, to cover the configuration space ($\mathcal{C}_{space}$), followed by local exploration using RRTs to achieve high roadmap connectivity. One example of such a method is SparkPRM [109], which employs PRM to quickly cover large areas of the planning space while RRT locally explores narrow passages.

2.2 Topological Properties in Motion Planning

There has been a growing interest in utilizing topological features to improve the performance of machines in various areas. These features can be extracted using mathematical concepts such as sheaf theory [31], persistent homology [49], Vietoris-Rips (VR) complexes [13], and landmarking approach [152]. Sheaf theory provides a framework for studying topological spaces that allows for the extraction of global features, and persistent homology is a tool for analyzing the evolution of topological features over different scales. The Rips complex is a family of simplicial complexes used to capture the topology of a point cloud, and the landmarking approach is a method for selecting a set of representative points that captures the topology of a space.

Topological features have shown benefits for object recognition in images, anomaly detection in time series data [102], and identifying homotopy classes of routes [101] in robotics. The notion of topological properties helps differentiate the path categories into homotopy classes, as discussed next.

Definition 2.1 ***Homotopy Class****: Two trajectories are said to be in the same homotopy class [20] if one can be smoothly deformed into the other without intersecting with obstacles. Otherwise, they belong to different homotopy classes. Figure 2.1 shows an example of different path classes.*

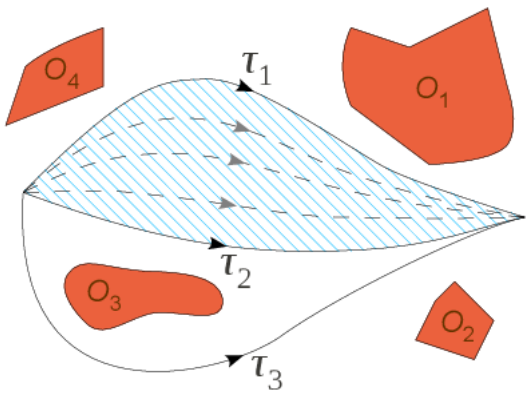


Figure 2.1: Paths τ_1 and τ_2 belong to same homotopy class, whereas τ_3 belong to different homotopy class.

The work in [21, 23, 24] represents homotopy classes of trajectories in 2D and 3D configuration space by using the line integral of "Obstacle Marker Function" over the path in

2D space or by exploiting laws from the theory of electromagnetism to routes through 3D objects with genus K (holes in the obstacles) to categorize pathways for K genus. The persistent homology notion used in [22] finds the homology class of trajectories most persistent for a given probability map. The work applies persistent homology to solve the fundamental problem of goal-directed path planning in an uncertain environment represented by a probability map.

Research in [100] considered homotopy classes of trajectories in general configuration space using Delaunay-Cech Complex filtration and abstracted the global information about routes using persistent homology. Another extension in [101] showed the application of a sampling-based approach to topological motion planning that is data-driven in nature. The work uses the Delaunay-Cech method to filter the data from the point-cloud dataset and improvises Dijkstra's algorithm to generate distance-vector terminology for a source vertex. Our research avoids using Delaunay-Cech Complex filtration due to the difficulty in computing complexes in high-dimensional $\mathcal{C}_{space}$, i.e., the curse of dimensionality.

$TARRT^*$ in [148] provides an efficient sampling structure for exploring a topological constraint over multiple homotopy classes. $TARRT^*$ enforces sampling that honors a set of possible homotopy classes and rewires the RRT^* tree to explore multiple homotopy classes in parallel. The topological information can either be assigned prior to the planning or queried during posterior path selection.

The online TIGRE algorithm in [55] solves the problem of autonomous robotic exploration. It builds a map with high-level information captured during online topological segmentation that incrementally generates an undirected connected graph of the environment. It formulates the exploration problem as the traversal of all the edges of the online roadmap. The work integrates Graph-SLAM features, contour-based topological segmentation, incremental graph construction, and online decision-making with an adaptation of a constrained depth-first search-based graph traversal algorithm for exploration tasks.

Although, the discussed methods provide ways to extract a topological description of the space and approximate sampling to achieve performance guarantees. It does not focus on the approximate measure of the extracted topological information needed to represent the planning space.

2.2.1 Topological Approaches

In recent years, the Vietoris-Rips complex has emerged as a promising alternative to the Čech complex. It is due to several advantages, including its ability to construct quick complexes from point clouds, avoid voids in the resulting complexes, and be easily implementable. As a result, researchers have increasingly turned to the Rips complex as a means of obtaining topologically correct approximations of metric spaces. For the sake of brevity, we refer to the Vietoris-Rips complex as the Rips complex.

The sparse Vietoris-Rips filtration method was designed to approximate the persistent homology of the Vietoris-Rips filtration in [107]. The approximate persistence diagrams computed at all scales showed the potential to make persistence-based methods tractable for much larger inputs. The persistence diagram of this new filtration did not deviate much from that of the Vietoris–Rips filtration and could also adapt to the Čech filtration. Shadow complexes introduced in [36] produce a projected shadow map from Rips complex to an n-dimensional Euclidean space that has its image captured with a more accurate approximation to the homotopy type of n-dimensional sampled space. More precisely, the projection map represents a simple planar graph of the Rips complex that accurately captures the connectivity for a planar point set in a free space. However, the map failed to preserve the higher-order topological data in dimensions greater than three. Another work in [152] proposed three different strategies, i.e., inductive approach, incremental approach, and maximal method, for faster computation of the Rips complex from a generated neighborhood graph of topological space. The maximal algorithm showed the most intriguing results by generating maximal cliques in a shorter time than the other two algorithms and provided a better approximation of the environment.

Some other research work approached the approximation of topological space by focusing on improving the accuracy of the Rips complex in the high-dimensional plane. One such study [13] presented mathematical proofs to show Rips complexes can provide topologically correct approximations of shapes for the notion of distances between points in the metric space. The work provides conditions under which the Rips complex of the point set at some scale reflects the homotopy type of the shape for a finite point set that samples a shape. For the infinite set of points in space, Čech and Rips complexes coincidentally degenerate to a trivial form and, thus, the hypothesis was stable only under small metric perturbation.

Another study showed the application of metric graph gluing in topological analysis of metric spaces [4]. It proved that the wedge sum of Vietoris-Rips (resp. $\check{C}$ech) complexes is homotopy equivalent to the corresponding complex for the metric wedge sum and generalized this result in the case of Vietoris-Rips complexes for certain metric space gluing. The results in this paper constituted a step towards understanding the greatest extent possible for the topological structure of a large class of metric graphs via persistent homology. It provides a characterization of the persistence profiles of metric graphs obtained via certain types of metric gluing. But the work limits its application to two-dimensional vertices of path.

In this research, we apply and extend the notion of the Rips complex to acquire the topological approximations of $\mathcal{C}_{space}$ and use its metric gluing property to incrementally plan a path in difficult and complex regions of the environment.

2.3 Geometric Properties in Motion Planning

Finding a safe and optimal path in the presence of obstacles has been an important research topic in robot motion planning. Recall that the geometric features define the shape of an object using critical points information, i.e., local minima, local maxima, etc. Planning a path or motion near obstacles becomes possible by leveraging the geometric properties of the $\mathcal{C}_{space}$. Many researchers have employed the advantage of mathematical tools to improve the quality of paths by sampling-based planners.

Early research work [71] in the 90s presented an online motion planning algorithm for right-handed manipulators in 2D $\mathcal{C}_{space}$. The work identifies critical points on the boundaries of obstacles in $\mathcal{C}_{space}$ using line-of-sight and wall-following methods. Work in [12, 3, 2] proposed an exact cellular decomposition for the coverage task. It used Morse function critical points to partition the space into cells such that the structure of each cell enables a planner to use simple control strategies such as back-and-forth motions for coverage tasks. It modified the complete coverage problem to an incremental construction of a graph representation that encodes the topology of the Morse decomposition with changes occurring at the critical values. Thus, the method guaranteed coverage in an unknown space.

Crawling Probabilistic RoadMap (CPRM) [110] was introduced to perform real-time motion planning in the configuration space. The work defined a varying potential field f on

∂O as a Morse function where O is the obstacle in $\mathcal{C}_{space}$ and combined it with the naive PRM algorithm to reduce the computation time needed. CPRM is applicable for mechanisms whose $\mathcal{C}_{space}$ are algebraically known.

Motion planning explorer proposed in [94, 93] visualizes the local minima tree of a path that is invariant under minimization of a cost function (e.g., minimum-length, minimum-energy, or maximum-clearance). Using the Morse theory, the work defined a local minimum and utilized a fiber bundle construction (i.e., a sequence of admissible lower-dimensional projections) to organize them into a tree. Based on the user's input on the specification of a fiber bundle for each robot, the algorithm provided a tree with lower-dimensional projections allowing the user to visualize, debug and interact with a planning problem. Our developed approach differs by applying discrete Morse theory [56] to the extracted topological information that aids in overcoming the differential topology constraints for high-dimensional robots.

2.3.1 Coverage Path Planning

Coverage path planning determines a route that guarantees an agent will pass over every point in a given environment. It also requires that the algorithm covers the curvatures of the underlying space. Two new algorithms proposed for coverage path planning in agricultural fields uses a top-down approach to split the high-dimensional structure of the field and a bottom-up approach to cover the environment using prediction and brute-force methods [92]. A survey in [57] discussed several techniques that produce optimal coverage paths in planar spaces and reduce localization errors during coverage. Optimal coverage methods used numerous strategies such as line-sweep, genetic algorithm, and classical Morse decomposition. However, these techniques fail in their performance when processing a vast space. In this research, we achieve the near-optimal properties by preserving the essential topological and geometric information of $\mathcal{C}_{space}$ making it robust to varying space sizes (large or small).

Coverage planning algorithm with adaptive viewpoint sampling constructs accurate 3D models of large complex structures using Unmanned Aerial Vehicles (UAV) for inspection purposes [8]. The integrated sensor models generate a coverage path offline to compute the prediction of the coverage percentage. A study in [33] presented a survey on coverage path

planning with UAVs, addressing simple geometric flight patterns, such as back-and-forth and spiral, and more complex grid-based solutions considering full and partial information about the area of interest. The study showed that coverage planning methods focused on minimizing the path length, the mission execution time, and the number of turning maneuvers to save energy. However, in some cases, the performance metrics such as path length and energy consumption conflicted with scenarios where shorter paths contained more abrupt maneuvers leading to more energy consumption.

The discussed methods provide maximum coverage of the underlying space while reducing the robot's energy consumption. In this research, we provide a roadmap that covers all sub-regions of the $\mathcal{C}_{free}$, i.e., it encloses the entire $\mathcal{C}_{free}$ with samples at a safer distance from $\mathcal{C}_{obst}$. The roadmap is a complete coverage graph of the $\mathcal{C}_{space}$ representing a homotopy-equivalent topology of it. As a result, our roadmap generated with reduced computation time and low memory usage inferred low energy consumption during robot navigation.

2.4 Path Diversity in Motion Planning

Some real-world applications emphasize the need for diverse paths, e.g., route planning [38], navigation of mobile robots among dynamic obstacles [95], protein folding [79] where different fold configurations of the protein mean distinct function representations, etc. We next discuss some past research work presented in this area.

Path diversity introduced in [70] defines a path set as a collection of feasible pathways and their corresponding control sequences. A path set contains high path diversity if there is an improved performance in the presence of obstacles and goal-seeking behaviors. Having a variety of path options is essential. When a robot or self-driving car has access to a motion planning algorithm that can generate different paths, it reduces the computational complexity and the need for re-planning by giving more flexibility and adaptability in different situations. Path diversity lends itself naturally to the dynamic planning paradigm, which means having different pathways that lead to the same goal determines that if one path becomes inaccessible due to a sudden appearance of obstacles, alternative routes still exist in the set and, thus, the need for re-planning gets reduced. Research has more recently gone into formalizing path diversity and the best ways to characterize and produce them during a path planning event.

A $\mathcal{C}_{space}$ diverse path planner proposed in [135] returns low-cost distinct paths from a graph by calculating the distance between path curves using the Fréchet distance measure. The method tunes two parameters, i.e., branching factor and ball radius, to retrieve diverse paths from a sparse roadmap. A suitable balance of values for the branching factor and ball radius is needed to produce trajectories without exhausting the queue or removing a vast collection of edges along and near the ancestor path. The filtering criteria reject routes that are either too close to a previously accepted pathway or too long.

For a graph with a broad explicit set of paths, work in [30] provided two approximate algorithms (Inner-Product and Inclusion-Exclusion) that pruned large sets of candidate paths or trajectories down to smaller subsets while maintaining desirable characteristics in terms of overall reachability and path length. The methods discretized the environment into square segments and then greedily chose paths that minimized the number of squares the trajectories hold in common. Its extension in [52] further analyzed the survivability of the pathways from the collection of diverse routes w.r.t. random placement of the robot into an obstacle field in a static or dynamic space.

To simplify the problem of diverse path planning for high-dimensional robots, the method in [134] iteratively found different paths in the $\mathcal{C}_{space}$ for a simplified multiple path problem by reducing the size of the robot (called inhibited regions). Although their method provided an increased success rate for new solutions, the oversimplification of the robot's dimension reduced the robustness of their approach. One key difference with our method is that our graphs originate from the $\mathcal{C}_{space}$, where we capture the necessary topology information like corners of obstacles, boundary information, and nearness of vertices regardless of how they are connected in the graph. This is crucial because local changes in the planning space can affect multiple vertices close to each other in the $\mathcal{C}_{space}$. As a result, our approach maximizes the use of non-degenerate critical points to compute and distinguish diverse paths in the environment.

2.5 Motion Planning for Articulated linkage robots

The ability to automatically plan a motion task given geometric models of the manipulator (articulated linkage robot) is critical for redundant manipulators. However, motion planning failures are common due to the PSPACE completeness of the motion planning prob-

lem and complexity of the planning environments [67]. To address these issues, researchers use two main approaches: 1) causality analysis, which identifies the causes of planning failures and suggests recovery actions [9]; and 2) avoidance, which leverages domain or historical information to classify planning spaces and problems as easy or difficult [113].

Minimum Constraint Removal (MCR) problem presented in [62] aims to remove the fewest geometric constraints necessary to enable a collision-free path between the start and goal configurations. It proves that determining the minimum number of restrictions needed to resolve a failed planning problem is an NP-hard problem. The work presents a sampling-based algorithm with greedy heuristics to cluster connected configurations in each region of the obstacle partition to compute parsimonious explanations for path planning failures. In comparison, our research utilizes node visibility and edge expansiveness to narrow down the re-configuration search set and prioritize the topological information for faster re-computation of relative paths for fault-tolerant path planning. Our planner can be used, in conjunction with the MCR problem, to provide a solution by re-using the roadmap information once the constraint is analyzed.

The Combined Task and Motion Planning (CTAMP) problem introduced in [75] refers to finding a geometrically feasible plan by pointing out a culprit detection problem at the interface between the symbolic and geometric search spaces. The challenge of isolating the minimal number of factors explaining the failure is the culprit detection problem. The work proposes two techniques to address it. The first method is a geometric reasoner that computes minimal explanations for faults occurring in the process of geometrically instantiating a symbolic sequence of actions. The method detects culprits in a constraint network with the relaxed version of the geometric part of CTAMP. The second algorithm involves constructing a graph of geometric dependencies between the actions of unfeasible symbolic plans to extract sub-sequences of activities as potential culprit sub-sequences.

More recently, the work in [16] proposed a fault-tolerant control scheme for robot manipulators based on active inference of sensory fault. The proposed solution uses the sensory prediction errors in the free energy to generate residuals and thresholds for fault detection and isolation of sensory faults. While it succeeds in achieving the recovery by reducing the precision of the faulty sensor to zero, the method has difficulty extending past 2-dimensional robotic manipulators.

Some other approaches perform fault tolerance planning [131, 81, 132] by classifying and analyzing the planning space problems based on historical or domain knowledge. One such approach is an active fault-tolerant control scheme that uses a neural network-based fault diagnosis module and a reinforcement learning-based fault-tolerant control module to produce compensation torques to guarantee the system's safety and maintain control performance [146]. A Support Vector Machine (SVM)-based fuzzy back-stepping variable structure controller proposed in [98] diagnoses the fault and fine-tunes the variable structure coefficients to reduce the effect of failures on the robot manipulator. While these methods successfully detect system faults, they require extensive supervised training to identify failures efficiently.

In this research, we do not focus on learning-based fault detection and instead use the minimum path violation framework to find a feasible path in the changed $\mathcal{C}_{space}$. We define a "change" in the $\mathcal{C}_{space}$ as the inclusion or exclusion of $\mathcal{C}_{obst}$ that invalidates the existing solution.

2.6 Topology and Computational Biology

A particular domain of molecular modeling relates to predicting the binding conformation (configuration) for protein-ligand or protein-protein complexes; this problem is usually addressed with computational methods. These methods are required to accurately predict the 3D structure of the bio-molecule upon binding to the target receptor. A new research area has tried applying robotics-based motion planning techniques to this problem [6, 133, 5, 53], where it randomly samples alternative conformations, in consideration of the position and orientation of the bio-molecule inside the receptor's binding cleft and plans a feasible path to the binding site. We leverage the emerging application of robotics-inspired methods in computational biology to apply our method to study and examine the association behavior of bio-molecules using our method-identified features. An extensive discussion is available in chapter 5.

In the following sections, we discuss some previous research works that focused on the experimental evaluation of binding behavior during protein-ligand interaction, protein-protein interaction, or interaction with Intrinsically Disordered Proteins (IDPs).

2.6.1 Protein-ligand Interactions

Proteins do not function in isolation, and interactions help reveal vital functions and properties. However, the efficiency of these interactions depends on the dynamic and kinetic features of any pairs. Protein-ligand interactions are often analyzed using a theoretical (physics-based) or statistical (knowledge-based) approach. A ligand is any molecule or atom which binds reversibly to a protein to form a coordination complex. The geometric features exploration of the surface of a protein molecule enhances the understanding of molecular morphology and molecular mechanism that allows significant applications to drug design and protein-ligand interactions.

A hypersurface function in [17] demonstrates minimal molecular surfaces of bio-molecules that deal with internal and open cavities of the biomolecular structure to avoid geometric singularities for theoretical modeling of bio-molecules. The variational multiscale strategy in [54] combined the geometric and physical modeling of a biomolecular structure in the Lagrangian representation to study the protein-ligand interaction. Another approach [144] used cartesian representation to evaluate the second-order differential of six different 3D structures of the biomolecules. It identifies the structure curvature descriptors to help predict good protein-ligand binding sites. However, the evaluation metric of the geometric properties proved expensive, limiting the robustness in the interaction analysis of macromolecules, e.g., proteins, membranes, DNAs, and RNAs.

Geometric features often involve too much structural detail and are frequently computationally intractable for biological macromolecule data sets [34]. Our research addresses the problem by efficiently extracting and processing protein geometric features using the discrete Morse function. These features provide maximum and minimum curvatures information of the protein surface that is beneficial in scouring geometrically favorable protein-ligand interaction binding sites.

2.6.2 Protein-Protein Interactions

An important area of study includes understanding how a protein binds to another protein's active site and what structural changes both molecules undergo during docking to the active site or its exit from it. Such information allows for predicting the possibility of an association between protein-protein pairs, the strength of their association, and the protein

activity level.

Protein function evaluation is a challenging task approached by various sequence-based and structural-based methods [10]. However, the fact that the function of a protein is intrinsically related to its 3D conformation (more than to its primary sequence) motivates the use of structure in predicting protein function [111, 128]. During protein-protein interactions, the geometrical structure of the underlying topological manifold play crucial roles that affect specific biologically related functions, such as driving the cellular immune response [104]. To this end, various developed computational approaches predict the 3D conformation for molecular docking [83, 85, 32, 89], where the bio-molecules bind to the protein regions with potential coherence of matching (concave) curvatures.

Several rigid-body docking techniques have emerged as helpful tools to assess the prediction of possible interacting pose between two protein bio-molecules for global docking [117, 45, 112]. These docking servers sample conformations of the smaller protein bio-molecule around the larger one and use scoring functions to determine the top docking predictions. AutoDock-based incremental protocol (DINC) [46] addresses the limitations of AutoDock's standard protocol by enabling improved docking of large bio-molecules. DINC performs docking using AutoDock incrementally instead of in one single step by dividing the docking problem into smaller sub-problems. HawkDock [136] integrates the rigid-body docking protocol of ATTRACT [42] docking algorithm to predict several binding poses and determines the near-native docking using HawkRank score. HDOCK [145] is a hybrid docking algorithm that combines template-based modeling and template-free docking. The method overcomes misleading templates by switching to a template-free docking protocol and calculates the docking energy score using a knowledge-based iterative scoring function. However, these docking servers are limited to the number of residues or the size of the receptors, which results in failure or degradation of their performance. Our method overcomes this limitation by focusing on the features of the protein surface model independent of its size.

2.6.3 Biological mechanism of Intrinsically Disordered Proteins (IDPs)

Intrinsically Disordered Proteins (IDPs) are involved in many biological processes, such as cell regulation and signaling, and their malfunction gets linked to severe pathologies [40, 14, 127]. Understanding the functional roles of IDPs requires studying their in-

teractions with other proteins, which is very challenging and needs a tight coupling of experimental and computational methods. In contrast to structured/globular proteins, it is not easy to represent IDPs by a single conformation, and their models require ensembles of conformations representing a distribution of states that the protein adopts in solution. Thus, investigation of IDP interaction with structured/globular proteins is indispensable for understanding many biological mechanisms [116]. In terms of applications, understanding such molecular interactions is essential for drug design in pharmacology or protein engineering in biotechnology.

Studying the conformation of highly dynamic IDPs is a challenge in structural biology [96]. Nuclear Magnetic Resonance (NMR), often used in the study of IDPs [65], is a versatile spectroscopy method for studying proteins that, importantly, do not require crystallization. However, NMR spectral data from IDP ensembles have provided conformational constraints. The NMR-constrained molecular dynamics (MD) [7] simulations need multiple copies of the protein (known as replicate exchange MD) to generate possible structural models which fail to ensure the validity of the result regardless of the method used to sample the conformations using NMR data. The work in [90] uses the NMR tool to characterize the structure and dynamics of IDPs in various functional states and environments. It describes the NMR parameters of the structural ensemble to quantify the conformational propensities of IDPs and the challenges associated with obtaining structural models of dynamic protein-protein complexes involving IDPs.

A method combined the molecular dynamics simulations with circuit topology (CT) to analyze the biological behavior of a human androgen receptor with a large N-terminal domain (AR-NTD) in [108]. The method constructed the circuit topology of a potentially charged bio-molecule to analyze the fluctuations in the chain using the root-mean-square-fluctuations (RMSF) and root-mean-square-deviations (RMSD) metrics. Although the interaction of IDPs with other bio-molecules is a critical problem that needs a good understanding of IDP's functionality for drug design, there is little effort devoted to investigating the behavior of IDPs using the surface properties of the binding protein. Through this research, we take the first step to evaluate the association behavior of IDPs through our algorithm by using the topological and geometric properties of the bio-molecular surface.

CHAPTER 3

Approximating the properties of the configuration space

In this chapter, we describe and discuss our approach to extracting topological and geometric information from the configuration space and its impact on robot motion planning problems.

3.1 Topological Feature Extraction

Our work in [126] introduced a new roadmap construction approach to preserve the topological properties of the underlying space. Our method first generates and connects samples to construct the Vietoris-Rips complex and, then, verifies a set of properties describing the *convexity* of the occupied subspace. These properties guarantee that the constructed Rips complexes are homotopy equivalent to the underlying space. It then trims the samples of the Rips complex using a sequence of simplicial collapses, resulting in a sparse roadmap representation of the $\mathcal{C}_{free}$ space that still captures the topological information of the environment while being sparse. Thus, the resulting set of samples gives a roadmap representation of the $\mathcal{C}_{free}$ space. Figure 3.1 shows the workflow of our approach.

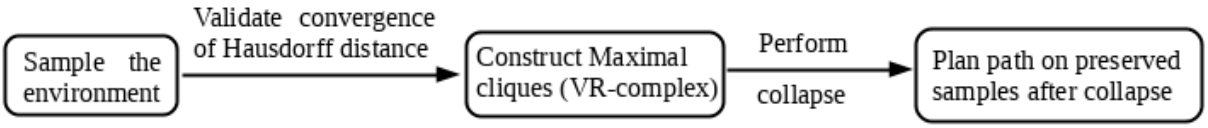


Figure 3.1: Workflow of topological data extraction

In the following sections, we define the mathematical concepts applied in this work to capture the topological information of the $\mathcal{C}_{space}$ and discuss some of its experimental results.

3.1.1 Mathematical Definitions

3.1.1.1 Abstract simplicial complex

An abstract simplicial complex K is a collection of subsets of a given set X closed under the subset operation. It is a generalization of a graph representing higher-than-pairwise

connectivity relationships. The elements of any set are called vertices, and the set itself is called a simplex. A simplex is a notion of the simplest possible polytope in any given dimension. For example, a 0-dimensional simplex is a point, a 1-dimensional simplex is a line segment, a 2-dimensional simplex is a triangle, and so on.

Definition 3.1 *(**Vietoris-Rips Complex**): Given a set X of points in Euclidean space E, the Rips complex $R(X)$ is the abstract simplicial complex whose K-simplices are determined by subsets of $K+1$ points in X with a diameter that is at most α. Figure 3.2 shows the Rips complex construction process as the value of α increases.*

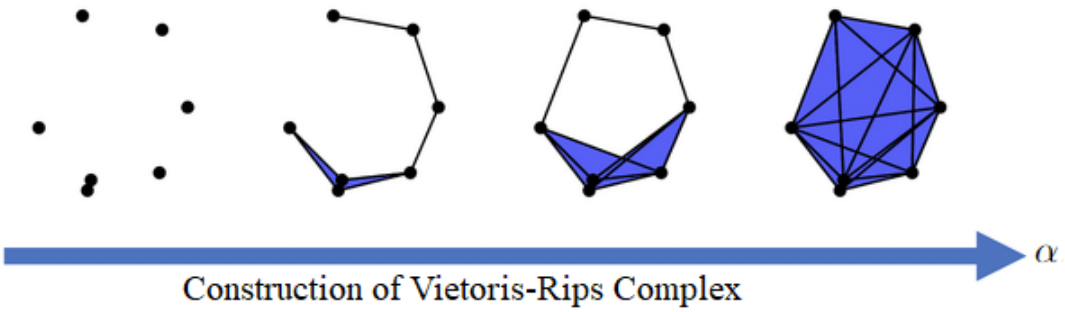


Figure 3.2: The points start forming connections as the α value increases, and the distance between two points is within the threshold α. It terminates the process when no more simplices are possible to construct, thus, resulting in a fully-connected graph.

Definition 3.2 *(**Čech Complex**): It is the abstract simplicial complex $C(X)$ whose K-simplices correspond to subsets of $K+1$ points that can be enclosed in a ball of radius $\alpha/2$, as shown in Figure 3.3.*

Topological thinning (simplicial collapse) [25] is an operation that shrinks simplicial complexes to homotopy-equivalent sub-complexes. This work applies simplicial collapses to reduce the complexity of simplexes through vertex deletion down to a core simplex (or skeleton) to maintain the topological structure of the configuration space by removing redundant information. The resulting roadmap represents a maximal clique graph of the $\mathcal{C}_{free}$ region.

3.1.1.2 Hausdorff Distance

The Hausdorff distance measures how far two subsets of a metric space are from each other [138]. In this work, we measure Hausdorff distance (ϵ) between set P – sampled robot

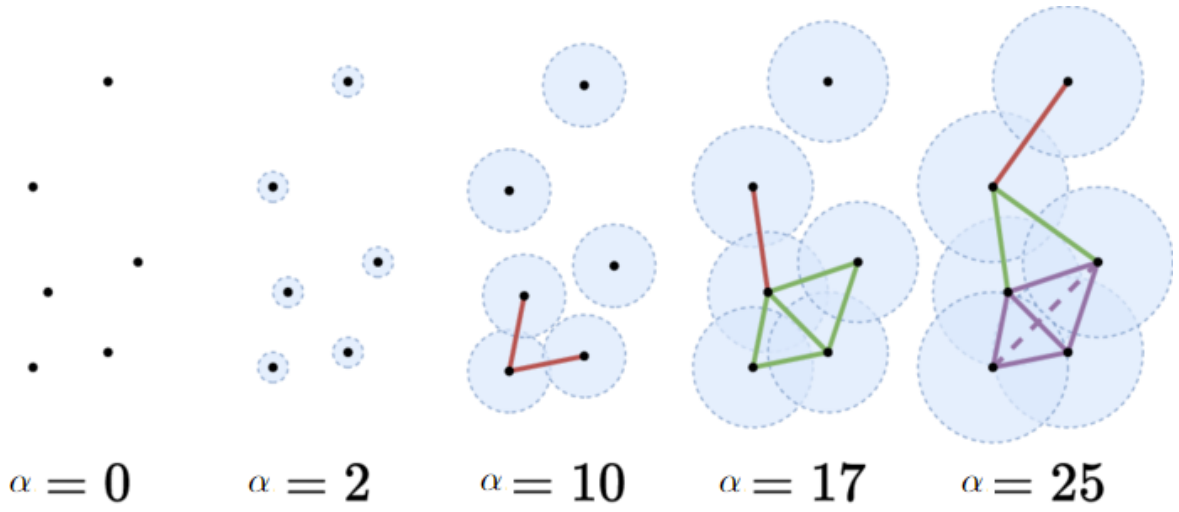


Figure 3.3: The figure shows Čech Complex construction for the value of α. The points start forming connections only when the balls of radius $\alpha/2$ centered at these points intersect.

configurations and set X – the $\mathcal{C}_{free}$ space. The algorithm uses a convex hull method to find the boundary points of set P to compute the closest distance ϵ between sample points and $\mathcal{C}_{space}$ boundary. The convex hull of a points set is the smallest convex polygon that encloses all of the configuration points in the set. In Figure 3.4, the blue line represents the boundary of the $\mathcal{C}_{space}$, and the green line is the boundary of the point cloud set P (calculated using convex hull). As the sample points get denser in the $\mathcal{C}_{space}$, the value of ϵ decreases and becomes constant above a definite sampling density.

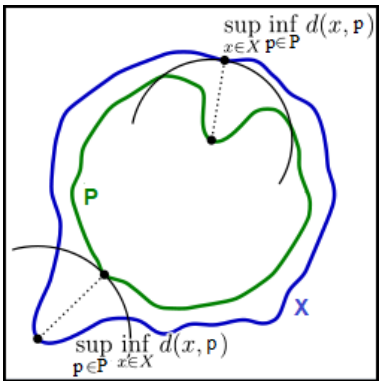


Figure 3.4: Hausdorff distance for set P and X

3.1.2 Algorithmic Foundation

We next describe the algorithmic foundation of our approach for memory-efficient roadmap construction using Rips complex and simplicial collapse.

3.1.2.1 From Rips complex to sampled-space topology

Generally, a Rips complex does not preserve the topology of the underlying sampled space. However, in [13], the authors showed that a Rips complex could be retracted to a Čech complex to approximate the topology of the underlying sampled space. Let us define the *flag complex* of a graph G, denoted Flag G as the maximal simplicial complex whose 1-skeleton is G. More precisely, this is the largest simplicial complex sharing with the Čech complex the same 1-skeleton. In addition, let us denote the Rips-complex $R(P,t)$ as the abstract simplicial complex whose k-simplices correspond to subsets of $k+1$ points in P with a diameter that is at most $2t$. The Čech complex $C(P,t)$ is the abstract simplicial complex whose k-simplices correspond to subsets of $k+1$ points that can be enclosed in a ball of radius t. Define α as an *inert value* of P if $\mathrm{Rad}(\delta) \neq \alpha$ for all non-empty subsets $\delta \subset P$. $\mathrm{Rad}(\delta)$ denotes the radius of $\mathrm{Hull}(\delta)$.

Then, given any point set $P \in \mathbb{R}^n$ and any real numbers $\alpha, \beta \geq 0$ with $\alpha \leq \beta$, define the flag complex of any graph G satisfying $R(P,\alpha) \subset \text{Flag } G \subset R(P,\beta)$ an (α,β)-quasi Rips complex of P. Also, let $v_n = \sqrt{\frac{2n}{n+1}}$. We follow the property in Theorem 7 from [13], which reads as follows.

Theorem 3.1 *Let $P \subset \mathbb{R}^n$ be a finite set of points. For any real numbers $\beta \geq \alpha \geq 0$ such that α is an inert value of P and $c_P(v_n\beta) < 2\alpha - v_n\beta$, there exists a sequence of collapses from any (α,β)-quasi Rips complex of P to the Čech complex $C(P,\alpha)$.*

The measure of convexity defects of X at a given scale is determined by the function c_p as given below.

$$c_p(t) = d_H(\mathrm{Centers}(X,t)|X) \tag{3.1}$$

Further, the graph can be shown to be homotopy equivalent to η-offset of the sampling space X, from Theorem 10 in [13], as follows.

Theorem 3.2 *Let ϵ, α and β be three non-negative real numbers such that $\alpha \leq \beta$ and $\eta = 2\alpha - v_n\beta - 2\epsilon > 0$. Let P be a finite set of points whose Hausdorff distance to a compact subset X is ϵ or less. Then, any (α, β)-quasi Rips complex of P is homotopy equivalent to the η-offset of X whenever α is an inert value of P and $h_X(v_n\beta + \epsilon) < 2\alpha - v_n\beta - 2\epsilon$.*

Here, Hull(X) denotes the convex hull of X, and

$$h_X(t) = d_H(\text{Hull}(X, t)|X) \tag{3.2}$$

$$Hull(X, t) = \bigcup_{\substack{\emptyset \neq \delta \subset X \\ \text{Rad}(\delta) < t}} \text{Hull}(\delta). \tag{3.3}$$

From the theorem, we can derive that in order to use a graph-like structure to approximate the homotopy of the sampling space we need to ensure sufficiently dense samples in $\mathcal{C}_{free}$ so that P is no more than ϵ away from the set X based on Hausdorff distance. Here, X is the set we would like to approximate using samples in P. Recall, Hausdorff distance $d_H(X, Y)$ is

$$\begin{aligned} d_H(X, Y) &= \max\{\sup_{x \in X} \inf_{y \in Y} d(x, y), \sup_{y \in Y} \inf_{x \in X} d(x, y)\} \\ d(y, X) &= \inf_{x \in X} d(y, x) \\ d_H(Y|X) &= \sup_{y \in Y} d(y, X) \end{aligned}$$

Therefore, if the samples P satisfy the above properties, we can construct a graph based on P and use the relations to approximate the underlying homotopy of X, even when the number of samples is small. In the case of the sampling-based motion planning approaches, where the connectivity is *guaranteed* when the number of samples reaches infinity, we can say our proposed method yields a bound on the number of samples. On the other hand, given a set of sample configurations P, we can also compute the relevant parameters to derive how much of the sample space X has the samples covered, where X can be $\mathcal{C}_{free}$ in the case of motion planning.

3.1.2.2 Roadmap construction

To construct the roadmaps that approximate the topology of the underlying configuration space, we first densely place samples that will satisfy the parameters presented in Theorem 3.2. We then use the samples to construct a Rips complex, which approximates the homotopy of the underlying $\mathcal{C}_{free}$ space. Finally, we perform simplicial collapse to remove the unnecessary samples from the Rips complexes to output a sparse roadmap.

The sampling process is similar to that of a PRM algorithm [68], with additional requirements, mainly the conditions mentioned in Theorem 3.1 and 3.2. During the sampling process, the Hausdorff distance are computed between the samples set P and set X, where X is the $\mathcal{C}_{free}$ space in our application. In [13], the authors stated that if the Hausdorff distance is smaller than the parameters measuring the space, the resulting point cloud (sampled points) approximates the topology of the space. Taking this into consideration (Theorem 3.2), we validate the expression $2\epsilon < 2\alpha - v_n\beta$, where $\beta = \alpha$ in our experiments. On verifying the sampling condition in the workspace, the output is a densely sampled $\mathcal{C}_{space}$ graph G.

3.1.2.3 Collapsing a Rips-complex

The convex hull of any nonempty subset of the $n+1$ points that define an n-simplex is called a face of the simplex (complex). A maximal face (facet) is any simplex in a complex that is *not* a face of any larger simplex. Given $\tau, \delta \in K$, if $\tau \subset \delta$, in particular $\dim \tau < \dim \delta$, and δ is a maximal face of K and no other maximal face of K contains τ, then τ is called a *free face*. A *simplicial collapse* of K denotes the removal of all pairs of simplices in γ such that $\tau \subseteq \gamma \subset \delta$. The work in [152] and [153] explains the equivalence between maximal faces in abstract simplicial complexes and maximal cliques in graph theory.

Given a simplicial complex K of dimension $n \geq d$, a d-skeleton of K is the subcomplex of K consisting of all the faces of K having dimensions at most d. Then, a graph representing the 1-skeleton of K can be referred to as the *underlying graph* and denoted as G_K. For brevity, we will refer to the 0-skeleton of K as vertices of G_K and 1-skeleton of K as edges of G_K. Then, we can derive the following results.

Lemma 3.1 *Given a complex K and its underlying graph G_K, let δ be a maximal face of*

K if a vertex v of G_K is a subset of δ ($v \subset \delta$) and no other maximal face of K contains v then there exist a sequence of simplicial collapses on K that can remove vertex v.

Proof: Let there exist a sequence of free faces $s_0, s_1, s_2, \ldots, s_m$, so that $s_0 \subset s_1 \subset s_2 \subset \ldots \subset s_m \subset \delta$ and $s_0 = v$. Let s_1 be one of the edges on G_K with v being one endpoint of the edge, let s_2 be the tetrahedron containing s_1, etc. Because each s_i is a free face, a simplicial collapse can remove it. Then, assuming the sequence of collapses start from s_m and moves towards s_0. Each collapse of s_i will not change the fact that s_i still is a free face of δ. Therefore, v is removable. □

Hence, we can extend the results to get the following theorem.

Theorem 3.3 *Given a complex K and its underlying graph G_K, let δ be a maximal face of K, and let V_s be the set of all the vertices v where v is a subset of δ and no other maximal face of K contains v. Then, after removing all vertices in V_s, there are no free faces in δ.*

Proof: Let us assume that after removing all vertices of V_s, at least one free face $\tau \subset \delta$ exists. If τ is of dimension 0, then it is a vertex that only belongs to δ, so it must have been part of V_s, so τ can not be of dimension 1 or above. If τ is of 1-dimension, i.e., an edge on G_K, then at least one vertex of the edge will belong only to δ. Otherwise, the edge cannot be a free face. Therefore, removing all vertices of V_s will remove this edge. Inductively, we can extend this to higher dimensions. Therefore, there cannot be any free face left once all the vertices in V_s get removed. □

3.1.2.4 Removing topologically unimportant vertices

Similar to the work in [152], Algorithm 1 first constructs a Rips complex using the maximal clique technique for faster computation. After generating the cliques, we transform the cliques' representation into binary form. Each node in the graph was represented based on the simplex to which it belongs. Given a graph with n nodes, the binary representation of a clique (or a sub-graph) is the binary string of length n in which the i^{th} character is 1 if the simplex (sub-graph) contains the i^{th} node and 0 otherwise. We then perform a bit-wise operation to find potential simplicial collapses using results from Lemma 3.1. As a result, it removes the vertices that are part of more than one clique after the operations.

Algorithm 1 Graph-Collapse(G)

Input: G: sampled graph from the point cloud; M: maximal clique, B: set of binary representation for each clique; T: set of vertices after topological collapse.

```
for all nodes in graph G do
    compute maximal clique M.
while M is not empty do
    for each clique in M do
        if node in clique then
            Set binary value '1' for node in B
        else
            Set binary value '0' for node in B
if B is not empty then
    T = B ⊕ B
    for each node in T do
        project node in graph G_new.
return G_new
```

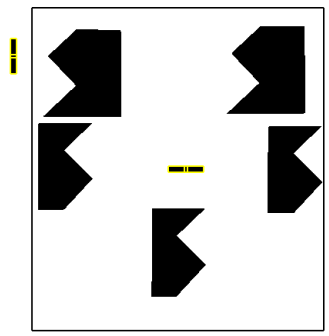
Finally, the algorithm returns a sampled graph G_{new} with vertices of non-colliding regions of $\mathcal{C}_{space}$, i.e., inclusive-maximal simplex. This resulting graph gives an approximate measure of the topological space (i.e., $\mathcal{C}_{free}$) in the $\mathcal{C}_{space}$.

3.1.3 Experimental Setup

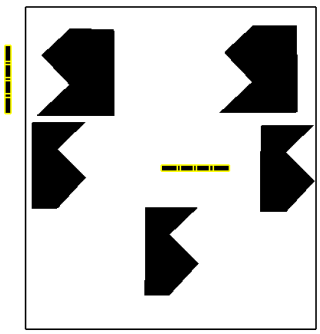
We executed all experiments on a Dell Optiplex 7040 desktop machine running the OpenSUSE operating system, and the code was developed in C++ using the PMPL library [74]. We performed experiments in three different environments, as shown in Figure 3.5, and generated samples ranging from 100 to 10,000 nodes. We used the brute force k-closest neighbor finding technique [88], the Euclidean distance metric, and a straight-line local planner for sampling and connection stages. We used the RAPID [60] collision detection method during the sampling, connection, and query stages.

- **ZigZag environment**: 2D environment with structured obstacles placed randomly as shown in Figure 3.5a and 3.5b. We tested for two configurations, one with a 2 DOF robot and one with a 4 DOF robot.
- **Heterogeneous 3D**: 3D maze environment with walls and narrow passages between the walls. A robot with a toroidal shape has to pass through maze-like tunnels to reach the goal, as shown in Figure 3.5c.

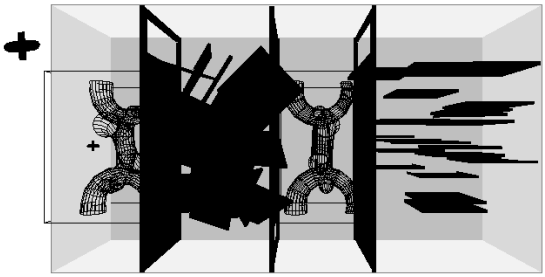
- **Helico**: A city representation with tall buildings and wires between buildings (Figure 3.5d). The robot is a rigid body representation of a helicopter and can change its vertical orientation based on the goal position.



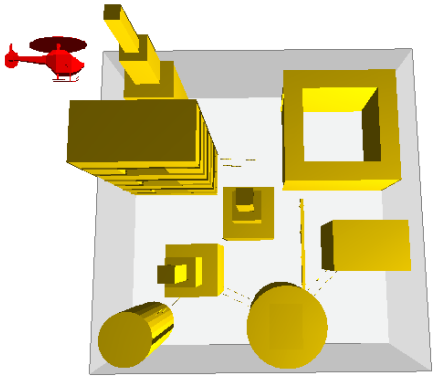
(a) **2 DOF ZigZag (snake-like)**



(b) **4 DOF ZigZag (snake-like)**



(c) **Heterogeneous 3D (toroidal plus)**



(d) **Helico (helicopter)**

Figure 3.5: Environments Studied

3.1.4 Results

3.1.4.1 Rips Complex computation

We performed preliminary experiments with two libraries that generate Rips complexes. We compared results to determine which library is most suited for our approach. We used the Rips-complex package of the GUDHI library [27] to construct simplicial complexes. The algorithm's time complexity is $O(v^2d + m^2d)$, where d is the dimension of the complex, v is the number of vertices, and m is the number of maximal simplices in the graph.

The Quick-cliques library [50, 51] generates maximal cliques using a modified Bron-Kerbosch algorithm by Tomita et al.[115]. We used a hybrid algorithm that applies a Rips-

complex approach to construct simplices. The algorithm's time complexity is $O(3^{d/3}nd)$ with n vertices and degeneracy d. Since Rips complexes, also known as clique complexes, the algorithm tries to generate maximal cliques as an output.

Table 3.1 and 3.2 compare results for GUDHI and Quick-cliques library in 2 DOF ZigZag environments with and without obstacles present. One can see that the Quick-cliques library computes maximal cliques faster than the GUDHI library as the number of nodes increases. Hence, we use the Quick-cliques library for the Rips complex construction.

Table 3.1: Constructing Rips complex in a 2 DOF ZigZag environment without obstacles

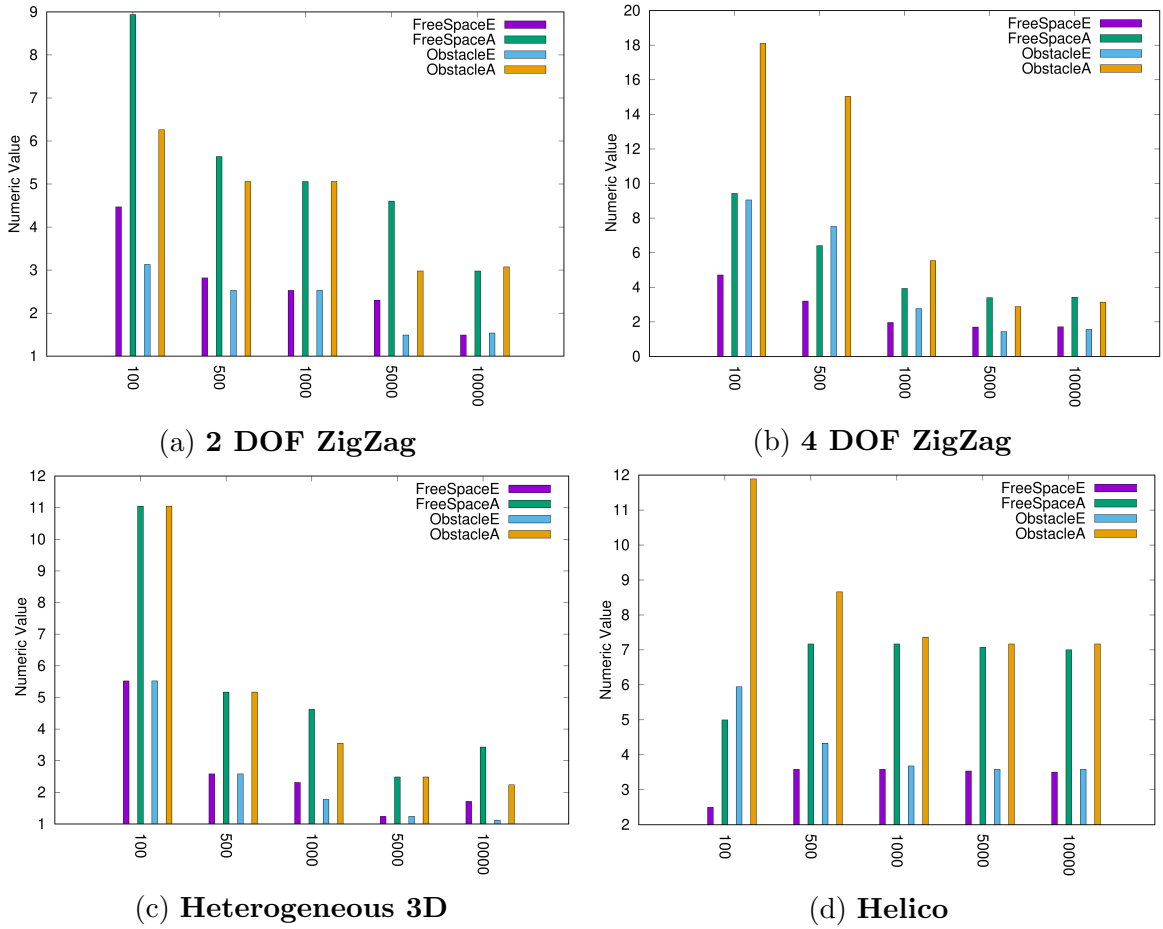
Library	**Number of Nodes**	**Cliques**	**Time taken (sec)**
GUDHI	100	210	**0.01**
Quick-Cliques	100	51	0.042274
GUDHI	10000	33552695	289.31
Quick-Cliques	10000	892190	**0.014901**

Table 3.2: Constructing Rips complex in 2 DOF ZigZag environment with obstacles

Library	**Number of Nodes**	**Cliques**	**Time taken (sec)**
GUDHI	100	268	**0.02**
Quick-Cliques	100	1378	0.042939
GUDHI	10000	81463172	675.6
Quick-Cliques	10000	1443062	**50.072735**

3.1.4.2 Sampling at different densities

We conducted two rounds of experiments on our three testbeds, one with obstacles and one without. We first performed experiments for the sampling conditions of P based on the $2\epsilon < 2\alpha - v_n\beta$; where $\beta = \alpha$, preconditions discussed previously in Section 3.1.2.2. Another condition, as illustrated in [13], states that as the sampling space becomes denser, the Hausdorff distance (ϵ) reduces or approaches a constant value. Once both conditions are satisfied, we construct a graph G of the densely sampled point cloud of the space and perform topology collapse to remove extraneous details. Our results show that the roadmap preserves the necessary topological information after the simplicial collapse, and the coverage of $\mathcal{C}_{space}$ is not compromised.



(a) **2 DOF ZigZag** (b) **4 DOF ZigZag**

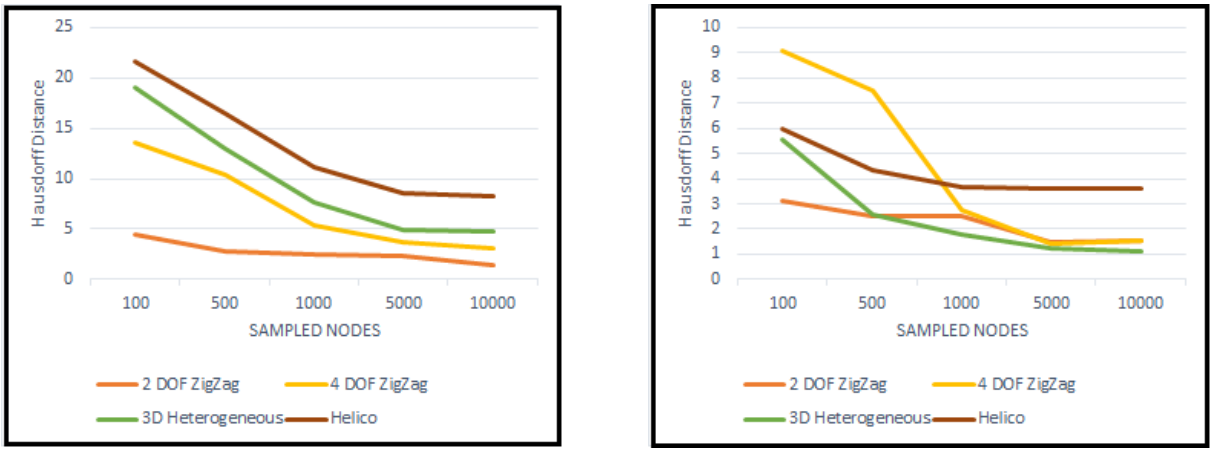
(c) **Heterogeneous 3D** (d) **Helico**

Figure 3.6: ϵ and α trends in obstacle and free environments

Sampling conditions: In Figures 3.6 and 3.7, we observed that the Hausdorff distance (ϵ) converges more gradually in an obstacle-free environment than in an environment with obstacles. The trend as shown in Figure 3.6 satisfies the conditions mentioned in [13], which states that the value of ϵ gradually decreases and becomes constant as the radius of the circle covering the $\mathcal{C}_{space}$ and the number of configurations increase. Here, we can deduce from the plots 3.6a to 3.6d that the value of ϵ decreases as the value of α and v_n increases. The purple and blue bars (2ϵ)(E) and the green and yellow bars $(2\alpha - v_n\beta)$ (A) in the histogram represented in Figure 3.6a to 3.6d show that in all cases both the pre-conditions are satisfied.

In the particular case of the Helico environment, as seen in Figure 3.6d, the ϵ value initially remains low and increases as the graph becomes denser before leveling off and becomes constant. The position of a robot in this environment is at the corner of the $\mathcal{C}_{space}$, so when the samples are computed initially, they are generated only near the boundary of the

$\mathcal{C}_{space}$, and, hence, ϵ value is low until the number of samples increases in the environment to produce better coverage. The ϵ values converge to constant as it reaches 10000 sampled nodes in all the planning space, as shown in Figure 3.7.



(a) $\mathcal{C}_{free}$ **environment** (b) $\mathcal{C}_{obst}$ **environment**

Figure 3.7: Convergence of Hausdorff distance in obstacle and free environments

Topology Collapse: Table 3.3 contains results for topology collapse experiments that utilize theorems and algorithms presented in Section 3.1.2.3 and 3.1.2.4. The results substantiate the ability to delete vertices, thus, confirming Lemma 1. We show a 40 to 90% reduction across all the environments, which indicates that our method can delete vertices while retaining the topological information of the space.

Table 3.3: Results after the Topology Collapse in the Free and Obstacle Environment

Environment	Nodes Before	Nodes After- Free	% Reduction	Nodes After- Obstacle	% Reduction
2DOF Zig Zag	10,000	5081	49.2	4826	51.7
4DOf Zig Zag	10,000	637	93.6	896	91.1
Heterogeneous 3D	10,000	4968	50.3	5061	49.3
Helico	10,000	5041	49.6	5023	49.8

Figure 3.8 gives a pictorial representation of one studied environment after the graph simplicial collapse. The sub-figures show the process of performing a topological collapse and getting a path for a simple robot. The figures indicate the following: (i) a $\mathcal{C}_{space}$ with obstacles, (ii) a 1-skeleton with five or few samples in $\mathcal{C}_{free}$, (iii) a 0-skeleton densely sampled graph, (iv) the structure that remains after the topological collapse, and (v) a successful path through the $\mathcal{C}_{space}$.

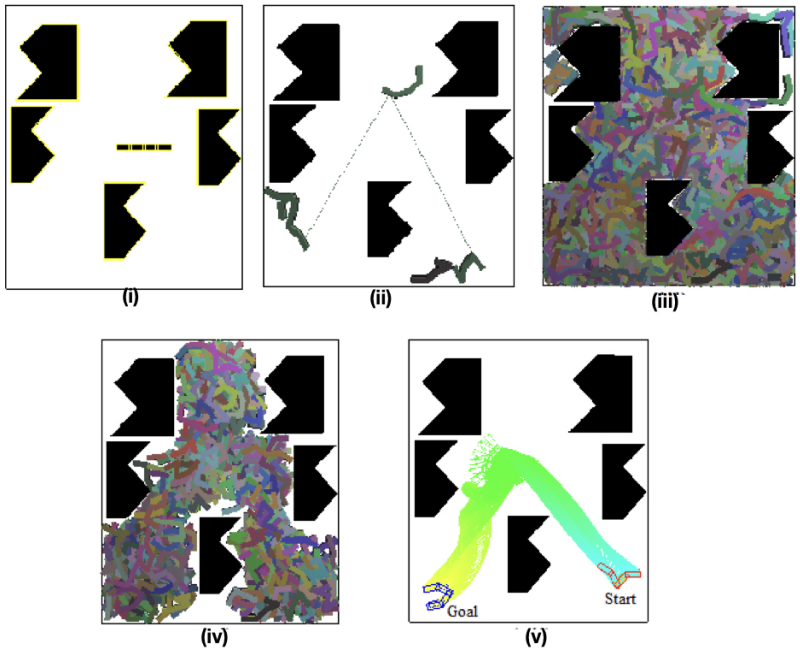


Figure 3.8: Phases of topological path planning

In addition, the samples generated to construct a Rips complex consists of locally complete subgraphs (cliques). After the collapse, the roadmap skips the edge information and captures only the vertices of a unique inclusive-maximal clique. Therefore, the memory needed to store the resulting roadmap scales linearly to the number of samples left after the collapse. Our constructed roadmap is comparable to k-nearest neighbor PRM roadmap but provides much richer topological information [152].

3.1.4.3 Planning with homology equivalent samples

Table 3.4 and 3.5 compare paths generated by PRM^* [67] using (i) the original point cloud and (ii) the vertices of the Rips-complex after the topological collapse in different environments in terms of total path cost and time needed to build a path. We report time to connect and query the environment alone to allow for fairness in our comparisons. The results show a significant improvement in all environments studied.

Overall, we observed that the reconstructed $\mathcal{C}_{space}$ provides an η-offset approximation of $\mathcal{C}_{free}$ region and has proven to be helpful in path planning while reducing the computation time and memory usage.

Table 3.4: Path planning time (in seconds) in the Free and Obstacle environments

$\mathcal{C}_{free}$ **Environment**	**Our Approach**	PRM^*	$\mathcal{C}_{obst}$ **Environment**	**Our Approach**	PRM^*
2DOF ZigZag	**62.1342**	229.922	2DOF ZigZag	**53.0861**	129.904
4DOf ZigZag	**64.7357**	11146.4	4DOf ZigZag	**2.52541**	24089.9
Heterogeneous 3D	**62.6602**	DNF	Heterogeneous 3D	DNF	DNF
Helico	**55.8919**	DNF	Helico	**58.4688**	82967

Table 3.5: Path planning cost in the Free and Obstacle environments

$\mathcal{C}_{free}$ **Environment**	**Our Approach**	PRM^*	$\mathcal{C}_{obst}$ **Environment**	**Our Approach**	PRM^*
2DOF ZigZag	**1003**	1438	2DOF ZigZag	**827**	1553
4DOf ZigZag	**916**	1324	4DOf ZigZag	**893**	1258
Heterogeneous 3D	**3714**	DNF	Heterogeneous 3D	DNF	DNF
Helico	**1806**	DNF	Helico	**1338**	2698

3.2 Geometric Feature Extraction

The work in [124] focuses on identifying the geometric features of $\mathcal{C}_{space}$ from the extracted topological information, as mentioned previously. Specifically, we define a discrete Morse function on the simplices of Vietoris-Rips complex that captures the local minima and maxima, i.e., critical points, of the obstacles' surface in $\mathcal{C}_{space}$.

3.2.1 Discrete Morse Theory (DMT)

Discrete Morse theory, originally defined by Forman [56], is a discrete analog of the classical smooth Morse theory. It is used in applications, such as ours, to simplify the topological information about the space by decreasing the size of the representation without affecting crucial properties such as the homotopy type. The context is either a simplicial or more general cellular complex.

Discrete Morse theory is a recent development in topology that has resulted in an explosion of applications in a wide range of fields. This paper is an example of such a robotics application. DMT should not be confused with discretized smooth Morse theory because it is a genuinely combinatorial subject and is thus amenable to efficient computer implementations. It is a theory that defines all of the trappings of smooth Morse theory and, so, can be used in place of the smooth theory. The benefits are enormous. It is purely a discrete theory, that is, applied directly to simplicial or more general cellular complexes. The ability to generate discrete functions from samples, such as the density-based function,

which is unavailable in a smooth context, is one of the many additional applications that can be particularly useful for our work. Before getting into the details of our approach, we will first illustrate the essence of the discrete Morse theory in a simple example, which includes the discovery of critical points crucial for this work.

3.2.1.1 Critical simplex in discrete Morse theory

Consider the simple but topologically non-trivial shape called the 2D torus as an obstacle model in the free space of 3D Euclidean space (shown in Fig. 3.9). It is impossible to embed the torus in the plane, but we can draw its planar triangulation by indicating parts of the boundary glued together when the labels match.

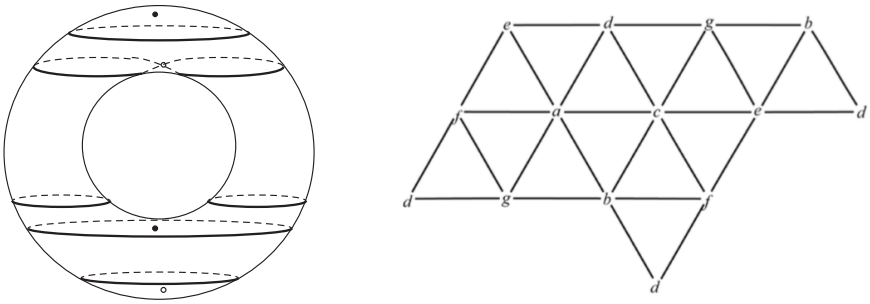


Figure 3.9: A planar triangulation of a Torus

Suppose we have meaningful assigned values to the vertices in this triangulation (shown in red in Fig. 3.10). We argue that these values can be extended to values on all simplices so that altogether we get a so-called *discrete Morse function*. Once a discrete Morse function

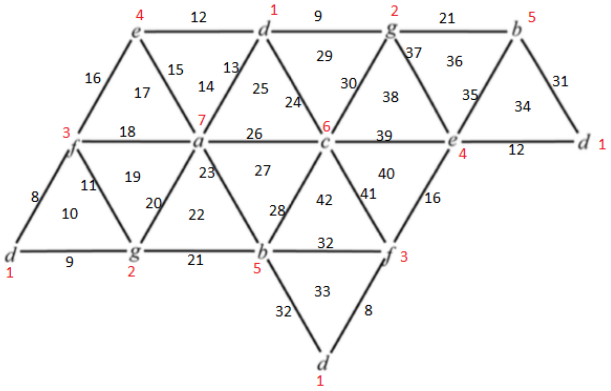


Figure 3.10: Assigned values on all simplices.

is defined, the theory gives the construction of a consistent flow indicated with arrows (see

Fig. 3.11). There are several critical cells of various dimensions but only one critical point d. As a result, we see that the discrete flow identifies the critical simplices. When the critical

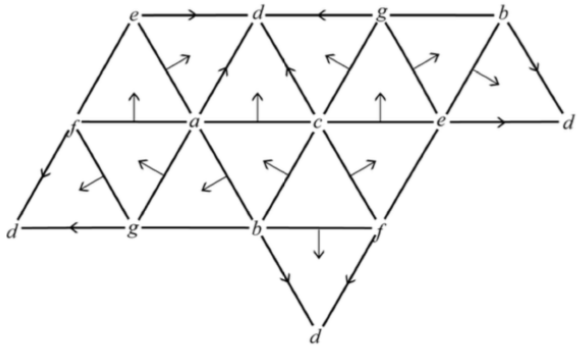


Figure 3.11: Identified critical point d.

simplex is 0-simplex, we refer to it as a critical point.

3.2.1.2 Constructing a discrete Morse function in $\mathcal{C}_{space}$

We adopt a discrete Morse theoretic view to enumerate the critical points of $\mathcal{C}_{obst}$. We define a density-based discrete Morse function on the constructed simplicial complex to extract the geometric property.

Definition 3.3 *(**Distance function**): Let $D(x)$ denote the distance between the point $x \in \mathcal{C}_{free}$ and the nearest point y on the closest obstacle $O_i \in \mathcal{C}_{obst}$, that is, $D(x) = \min_{y \in O_i} \|x - y\|$.*

Definition 3.4 *(**Density function**): Let $\Gamma(y, \varrho)$ be a density function where $\varrho > 0$ and y is the point on the obstacle surface. Our choice of the function Γ counts all neighbors in $R(X) \subseteq \mathcal{C}_{free}$ close to y within distance ϱ.*

Our function is, in fact, defined at any point in $\mathcal{C}_{space}$ and is given by

$$f(x) = D(x) \cdot \Gamma(y, \varrho). \tag{3.4}$$

Theorem 3.4 *(**discrete Morse function**): The restriction of f to the vertices of the Vietoris-Rips complex is the restriction of a discrete Morse function defined on all of the complex $R(\mathcal{C}_{space})$. The critical points of this function identify features of the obstacles.*

Proof: We first argue the second statement, then show how to obtain the Morse function. Let $\omega \geq 0$. We consider any closest obstacle O from the robot and the subset $X = D^{-1}([0, \omega])$. We denote $\text{Hull}(X, \omega)$ as the convex hull of the set X at scale ω from O, where t and s are the random vertices/points that define the elements in X. In other words,

$$\text{Hull}(X, \omega) = \bigcup_{t,s \in X} [t, s]. \tag{3.5}$$

Taking the obstacle O, the local maxima and minima of the function f occur on the surface of the obstacle O. They will be critical points of f in $\tau \subset O$ (as objects are assumed to be polyhedral) when $f \to 0$. The $\text{Hull}(\tau)$ determines the boundary of the obstacle surface containing these critical points. Let us take point $p \in X$. When $D(p) = 0$, that is $\omega \to 0$, the distance between point p and closest obstacle O becomes negligible. The density of neighboring points in $\mathcal{C}_{free}$ decreases on approaching closer to the obstacle O. However, the value of $\Gamma(y, \varrho)$ has a very meaningful limit. It is easy to see that the features on the surface of the polyhedron O can be detected as critical points of f on the $\text{Hull}(X, \omega)$. Thus, we determine them as critical points which satisfy the following conditions,

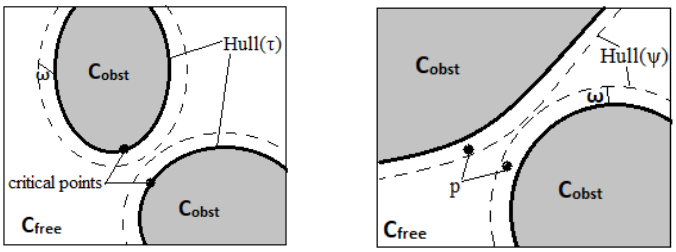
1. $\forall$ points $p \in X$; $\exists$ point $y \in O$ such that $f(p) \leq f(y)$,
2. $\forall$ points $y \in O$; $\exists$ point $p \in X$ such that $f(y) \geq f(p)$.

The identified critical point p will satisfy the equation

$$\lim_{\omega \to 0} f(p) = D(p) \cdot \Gamma(y, \omega). \tag{3.6}$$

Figure 3.12 shows an example of identified critical point on the obstacle surface.

To apply discrete Morse theory to identify the critical points of f restricted to $\text{Hull}(X, \omega)$, we represent f given on the vertex set of the Vietoris-Rips complex as the restriction of a discrete Morse function f defined on all of the complexes. It is a well-studied problem in discrete Morse theory with many solutions. □



(a) (b)

Figure 3.12: The figure shows the identified critical points on the boundaries of the $\mathcal{C}_{obst}$ in (a) and the feasible critical points (as denoted by p) in $\mathcal{C}_{free}$ closer to the critical points at a distance ω from the $\mathcal{C}_{obst}$, in (b).

3.2.2 Methodology

To extract the geometric information, we perform two associated steps. First, we abstract the topological representation of the $\mathcal{C}_{space}$ using the Rips complex. Second, we apply the discrete Morse function to this representation. Here, we describe implementation details to identify critical points and feasible critical points information for our topology map.

3.2.2.1 Feasible Critical Points in $\mathcal{C}_{space}$

Let $\Phi = D^{-1}([0, \varrho])$ be the compact set of points in $\mathcal{C}_{free}$ at most distance ϱ from the obstacle boundary such that the computed ϱ value can be given as

$$\varrho = \frac{1}{n} \sum D(s); \forall s \in \text{Hull}(R(X)), \tag{3.7}$$

where C denotes the set of identified critical points, $R(X)$ denotes simplicial complex vertices set, n is the cardinality of set C and $D(s) = \min_{y \in C} \|s - y\|$ from Def.3.3. The term clearance defines the offset distance of point p from the obstacle boundary, thus shifting p away from $\mathcal{C}_{obst}$ to $\mathcal{C}_{free}$. We say p becomes a feasible critical point with a maximum ϱ-clearance from the obstacle boundary if $p \in \mathcal{C}_{free}$. An example is shown in Figure 3.12(b).

For comparison, the Generalized Voronoi Diagram (GVD) provides a roadmap to extract high-clearance paths. The GVD defines the maximum clearance for a pathway from the $\mathcal{C}_{obst}$, as utilized in the Medial-Axis PRM [141]. An exact computation of the medial-axis distance is not practical for problems involving many DOFs (degrees of freedom) and a cluttered

environment with many obstacles, as this requires an expensive and intricate calculation of the $\mathcal{C}_{obst}$. We have proven in previous work that on reaching a needed sampling condition, the Vietoris-Rips complex provides a topologically equivalent representation of the space as the $\check{C}$ech complex. After the simplicial collapse, the resulting simplicial complex is a sparse sub-sampled graph that reconstructs the surfaces equivalent to the Delaunay complex, similar to those explained in [47]. Instead of computing the medial-axis distance from $\mathcal{C}_{obst}$ to the boundary of the Voronoi cell, we considered the closest distance from each critical point to the convex hull of the simplicial complex to find the clearance values from $\mathcal{C}_{obst}$. The minimum clearance is the average of all calculated offset distance values. In this work, we used the computed mean value as the clearance distance from the $\mathcal{C}_{obst}$, i.e., ϱ.

3.2.2.2 Generating ϱ-clearance samples in the $\mathcal{C}_{space}$

Algorithm 2 provides a roadmap with configurations at a distance ϱ from the obstacle in the $\mathcal{C}_{free}$, i.e., feasible critical points, by applying the discrete Morse function f on the constructed Rips complex S. The algorithm considers the convex hull of invalid nodes set as the $\mathcal{C}_{obst}$ boundary, returned by the function *GetObjectConvexHull* and computes the Morse values of these nodes to identify the potentially critical points on the $\mathcal{C}_{obst}$ surfaces, from equation (3.4). When the collapses in Rips complex S get close to the $\mathcal{C}_{obst}$ boundary, the distance becomes an equalizer making density the most important contributing factor. Thus, the local minima and maxima of f depend on the lowest and highest density value achieved by f at a point $y \in \mathcal{C}_{obst}$, in line 10. The algorithm inspects computed Morse values (in line 11) to identify the critical points for each $\mathcal{C}_{obst}$ in line 12. It captures the set of feasible critical simplices in $S \subseteq \mathcal{C}_{free}$ at clearance ϱ from the identified critical points in lines 14-15. The greater the value of ϱ decides the maximum clearance of a configuration from a $\mathcal{C}_{obst}$. Finally, the algorithm outputs a new graph G_{fcp} having configurations in the $\mathcal{C}_{free}$ at ϱ-clearance from the $\mathcal{C}_{obst}$ and the set of identified critical points of $\mathcal{C}_{obst}$ in C.

3.2.3 Experimental Setup

We perform experiments in simulation as a proof of concept of our methodology. The simulation experiments were performed in five different environments with robots ranging from 2 DOF to 14 DOF, as shown in Figures 3.5 and 3.13.

Algorithm 2 ϱ-clearance algorithm

Input: G: complete sampled graph from Sec.3.1; S: simplicial complex; O: Obstacle set, D: distance function, Γ: density function, f: Morse values set function, ϱ: value from eq.(3.7), C: set of identified critical points, N: set consisting of configurations around identified critical points, G_{fcp}: a graph of feasible critical points.

C $\leftarrow$ *null*, N $\leftarrow$ *null*;
G = *GenerateGraph()*; ◁ Refer Sec.3.1
O = *GetObjectConvexHull(G)*;
S = *CollapseComplex(G)*; ◁ Refer Sec.3.1
if S is not empty **then**
  for each obstacle $o_i \in O$ **do**
    for all sample $x \in S$ **do**
      $D(x) = min_{y \in o_i} ||x - y||$ ◁ Refer Def. 3.3
    for all node $y \in o_i$ **do**
      $\Gamma(y, \varrho) = \bigcup_{||x-y|| < \varrho} x \in G$ ◁ Refer Def. 3.4
      $f(y) = D(x) \cdot \Gamma(y, \varrho)$
      if $f'(y) \rightarrow 0$ **then**
        C = C $\sqcup$ y
  for all y $\in$ C **do**
    N = N $\sqcup$ $\Gamma(y, \varrho)$
for each neighbor n $\in$ N **do**
  $G_{fcp} \leftarrow$ N[n]
return $\{G_{fcp}, C\}$

- **2D environment**: The 2D space consists of a point robot with random obstacles in it, as shown in Figure 3.13a (referenced [66]).

- **Parking Garage**: The 3D environment has a vehicle parking garage structure, as shown in Figure 3.13e. The robot is a planar car with 3 DOF.

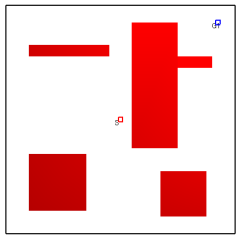
- **Urban environment**: This environment consists of buildings as obstacles in the city-like structure, as shown in Figure 3.13b. The robot is a 6 DOF drone.

- **Kuka YouBot environment**: The environment consists of a tree as an obstacle and a fixed base Kuka YouBot in it, as shown in Figure 3.13c. This robot is a simulation replica of Kuka YouBot [72] with an extended long arm (10 DOF). The robot moves its arm around the tree branches to grasp the fruit.

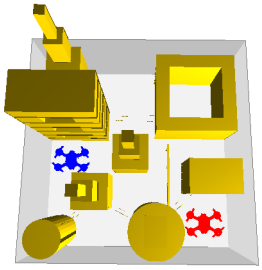
- **PR2 robot environment**: The environment consists of two pillar blocks placed on the table as obstacles where the robot is required to pass through the blocks to grasp

the stick kept on the other side, as shown in Figure 3.13d. The robot is a simulation replica of the PR2 robot [140] with only the right-hand arm (14 DOF) and a fixed base.

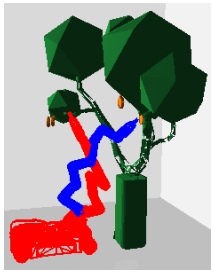
To ensure the accurate verification of our sampling condition, we integrated the computation of the reachable boundary volume of the robot as its environment boundary to meet the sampling condition. The computation time taken for identifying critical and feasible critical points for these robots is negligible, as empirically observed in Section 3.2.4.



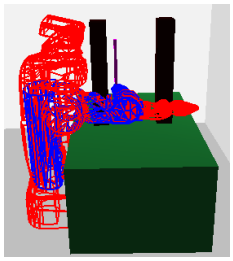
(a) 2D environment (square)



(b) Urban environment (drone)



(c) Kuka YouBot environment



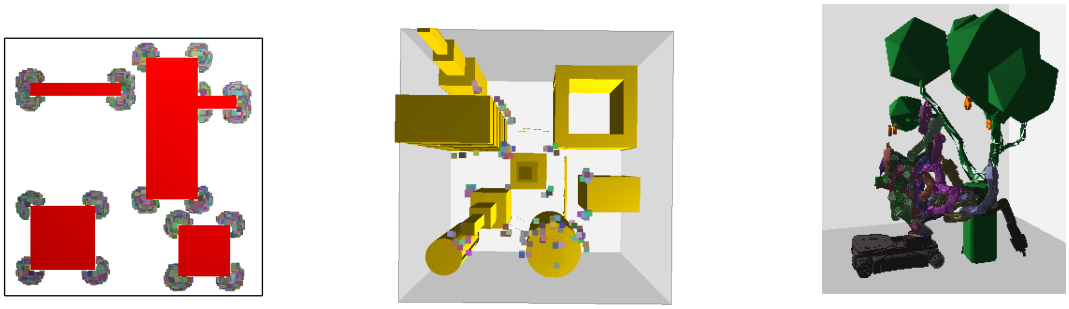
(d) PR2 robot environment

(e) Parking Garage (car)

Figure 3.13: From left to right, the robot has 2 DOF in (a), 3 DOF in (e), 6 DOF in (b), 10 DOF in (c), and 14 DOF in (d). The created testbeds are the simulation of the real-world robots and the environments to demonstrate the application of our algorithm.

3.2.4 Experimental Results

In this section, we discuss the results obtained using different PRM sampling strategies, i.e., Uniform [68], Gaussian [28] and Bridge-Test [64] planners, and RRT methods in 3D environments. We compared the results with two topology baseline RRT methods, Dynamic Domain RRT [147] and Dynamic Region-based RRT [44]. We also show the performance of the RRT, RRT^*, PRM, and PRM^* methods in the 2D environment (used by Karaman



(a) 2D environment (b) Urban environment (c) Kuka YouBot environment

Figure 3.14: The figure shows feasible critical points around the identified critical points of the $\mathcal{C}_{obst}$ for three different environments. In (b), a point-size view of the drone is shown for better visualization of feasible critical points. In (c), we see the feasible critical points of the robot are around the branches and bark of the tree.

et al., in [66]) and average the result values over 10 experiment runs in each case, i.e., evaluated total 500 trials for the 3D environments and 240 for the 2D environment. We used bi-directional RRT methods in all environments.

3.2.4.1 Topology Map

Table 3.6 shows the statistical result values computed for each sampler in each environment over ten runs. The column "Total Nodes" represents the generated density map nodes obtained on satisfying the sampling condition from [126]. In "% Reduction," we report the percentage of nodes removed after a simplicial collapse, and in "% FCP," we provide the percentile of feasible critical points present in the topology map. The topology map contains both the topological and geometric information of the $\mathcal{C}_{space}$, and its nodes are listed in the column "Extracted Nodes." We highlight the sampling strategy for each environment that preserved the topological and geometric information in their map with low memory overhead and improved the performance of existing RRT and PRM methods. Bridge-Test performed best in the narrow passage regions of the Parking Garage and PR2 robot environments, Uniform (PRM) performed best in the open space region of the Kuka YouBot environment, and Gaussian performed best in the cluttered region of the Urban environment. Our approach, like our previous machine learning method [118], revealed the core functionality of these sampling strategies, resulting in improved performance. So, the extracted geometric and topological properties of $\mathcal{C}_{space}$ aid in guiding the robot for memory-efficient path planning.

Figure 3.14 shows the captured feasible critical points for three different environments.

Table 3.6: STATISTICS OF EXTRACTED TOPOLOGICAL AND GEOMETRIC INFORMATION

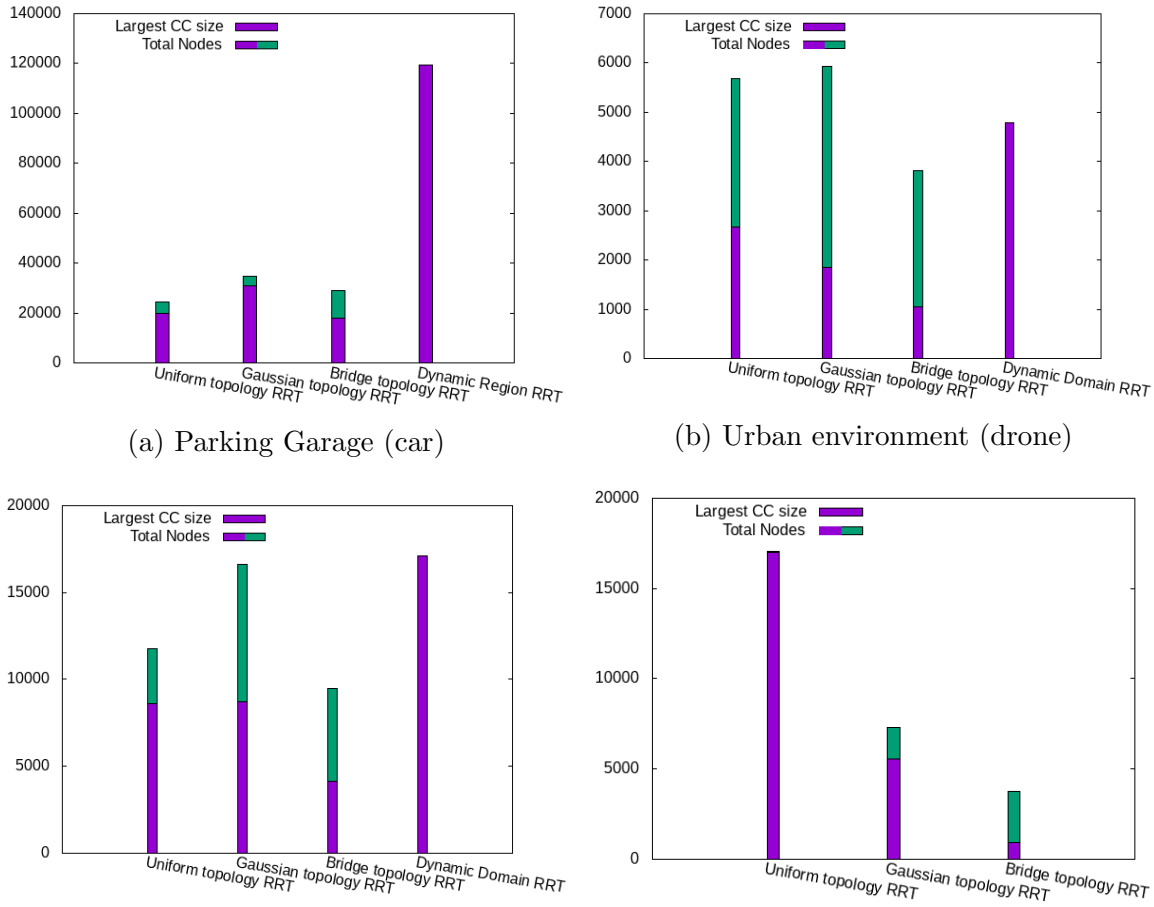
Environment	Sampler	Total Nodes	% Reduction	% FCP	Extracted Nodes
Parking Garage	Uniform	30000	40	20	18084
	Gaussian	30000	32	15	20616
	Bridge-Test	**25000**	**44**	**12**	**14124**
Urban environment	Uniform	10000	46	9	5455
	Gaussian	**10000**	**41**	**12**	**5872**
	Bridge-Test	5000	34	20	3287
Kuka YouBot environment	Uniform	**15000**	**26**	**46**	**11040**
	Gaussian	20000	27	47	14593
	Bridge-Test	10000	28	42	7156
PR2 robot environment	Uniform	20000	28	56	14466
	Gaussian	5000	36	19	3222
	Bridge-Test	**2000**	**53**	**1**	**940**

The survey in [57] concluded that when combined with topology methods, SBMP methods provide promising results for optimal coverage path planning. In this work, we show how our approach combines the topological, i.e., the homotopy equivalent map of the $\mathcal{C}_{free}$ space, and the geometric, i.e., the critical points and configurations near the $\mathcal{C}_{obst}$, properties into a topology map. This topology map contains enough nodes to cover all sub-regions of the $\mathcal{C}_{free}$ in the $\mathcal{C}_{space}$. Our method has usefulness in determining a path that can pass through all nodes in a given area or volume of interest while avoiding obstacles in coverage path planning problems. This work examines the quality of paths generated by a topology map (a $\mathcal{C}_{space}$ coverage map). Using our topology map, we demonstrate the convergence of the RRT and PRM methods to the near-optimal solution, i.e., produce paths within feasible bound from $\mathcal{C}_{obst}$ using a memory-efficient roadmap.

3.2.4.2 Comparison to RRT-based algorithms

We input the topology map generated by the Uniform, Gaussian, and Bridge-Test sampling strategies as an initial roadmap into the RRT method. Dynamic Domain RRT could not find a path for the Parking Garage environment, and Dynamic Region-based RRT was unsuccessful in finding a pathway for the Urban and Kuka YouBot environments. Both methods failed to complete in the PR2 robot environment. Our topology methods, on the other hand, finished planning paths in all four environments.

Nodes and largest clique size: Figure 3.15 depicts the total number of nodes in the roadmap as well as the size of the largest connected component (CC). The number of nodes in the largest CC does not equal the total number of nodes in the topology map, indicating that the number of nodes required to connect the start and goal configuration is less than the total number of nodes in the topology map. Dynamic Region-based RRT and Dynamic Domain RRT acquired all nodes from their roadmap to establish a path. Thus, our methods require fewer nodes than baseline methods to produce a pathway to the destination.



(a) Parking Garage (car)

(b) Urban environment (drone)

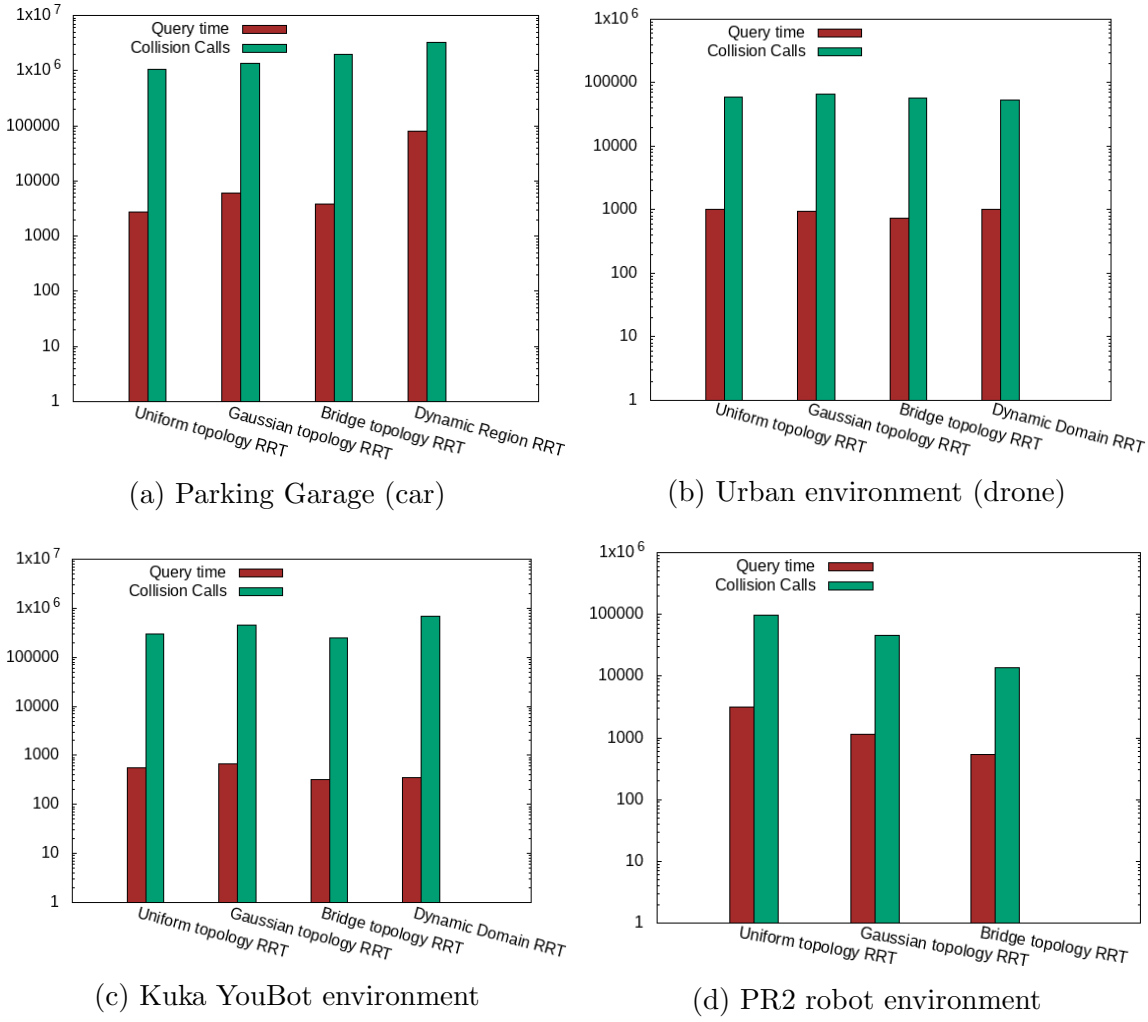
(c) Kuka YouBot environment

(d) PR2 robot environment

Figure 3.15: The figure shows the total number of nodes present in the roadmap in the purple-green bar and the size of the largest connected component (CC) in the purple bar for all environments. The size of the largest CC indicates the number of nodes connected to establish a path from start to goal position by RRT planners.

Query time and collision calls: Figure 3.16 depicts the total time required to solve a query and the number of collision check calls made. The query time accounts for topology

map generation time, node connection time duration, and path planning time. The collision calls count the number of times nodes or edges are validated, and we examined their impact on the topology map's path planning time. Our planners used fewer collision checks, resulting in less query time in all three environments than the baseline methods. In the PR2 robot environment, BridgeTopologyRRT outperformed other methods.



(a) Parking Garage (car)

(b) Urban environment (drone)

(c) Kuka YouBot environment

(d) PR2 robot environment

Figure 3.16: The figure shows the total query time and the number of collision calls invoked for all RRT methods.

Path Quality: Table 3.7 depicts an improvement in path quality after employing topology maps. We discovered that BridgeTopologyRRT produced shorter paths in Parking Garage and PR2 robot environments as explained in Section 3.2.4.1. Similar outcomes were observed for UniformTopologyRRT in the Kuka YouBot environment and GaussianTopologyRRT in

the Urban environment. As a result, we can conclude that our topology methods can produce shorter paths than baseline methods, with less computation time and fewer nodes.

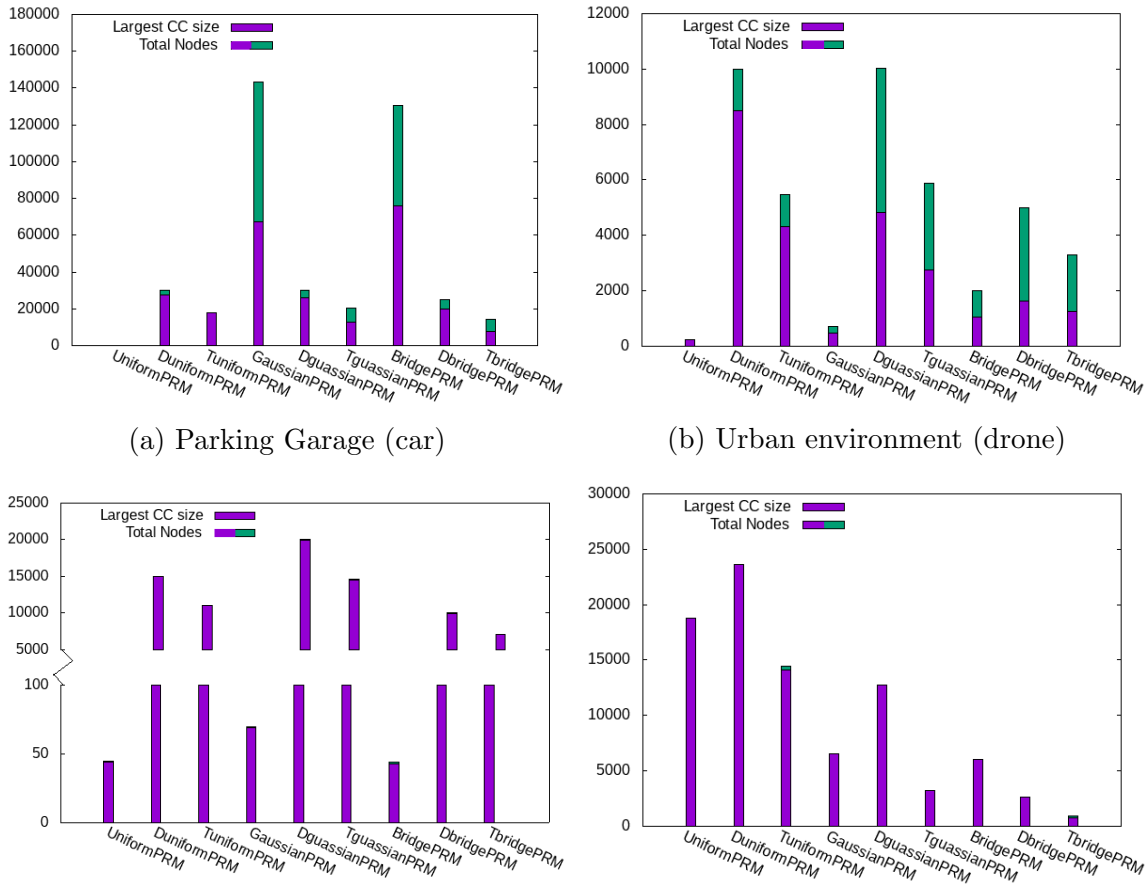
Table 3.7: Path cost achieved by different RRT planners

Methods	**Parking Garage**	**Urban**	**Kuka YouBot**	**PR2 robot**
Uniform Topology RRT	311.10	173.08	**14.03**	2.09
Gaussian Topology RRT	367.78	**164.37**	16.61	2.27
Bridge Topology RRT	**288.3**	184.5	21.56	**1.86**
Dynamic Domain RRT	N/A	167.15	23.39	N/A
Dynamic Region-based RRT	364.7	N/A	N/A	N/A

3.2.4.3 Comparison to PRM-based algorithms

As initial roadmaps, we input the PRM method with two different $\mathcal{C}_{space}$ coverage maps: 1. a density map - nodes generated to provide complete coverage of $\mathcal{C}_{space}$ and 2. a topology map - nodes preserved from the density map after the collapse and geometric feature extraction. We compare the performance of PRM with density map-based PRM and topology map-based PRM for all three samplers in all four environments. In the plot labels, we indicated density map methods with the prefix 'D' and topology map methods with the prefix 'T.'

Nodes and largest clique size: In Figure 3.17, we noticed that our topology map captured nodes in regions isolated from open free space, i.e., a space enclosed by $\mathcal{C}_{obst}$. Given this, we can conclude that, regardless of the sampler method used, our approach aided in capturing nodes close to $\mathcal{C}_{obst}$, covering enclosed sub-regions of $\mathcal{C}_{free}$, in the same way, that an obstacle-based sampler captures nodes close to $\mathcal{C}_{obst}$. Overall, we conclude that our topology map-based methods required fewer nodes to plan paths in Parking Garage and PR2 environments than other methods. However, in the Kuka YouBot and PR2 robot environments, all nodes were connected to form one largest CC for all planners, as shown in Figures 3.17c and 3.17d. Because the absence of enclosed regions in these environments makes the planners use all nodes from their roadmap for the query analysis. After 400 hours of continuous running, the uniform sampling strategy failed to find a path in the Parking Garage environment.



(a) Parking Garage (car)

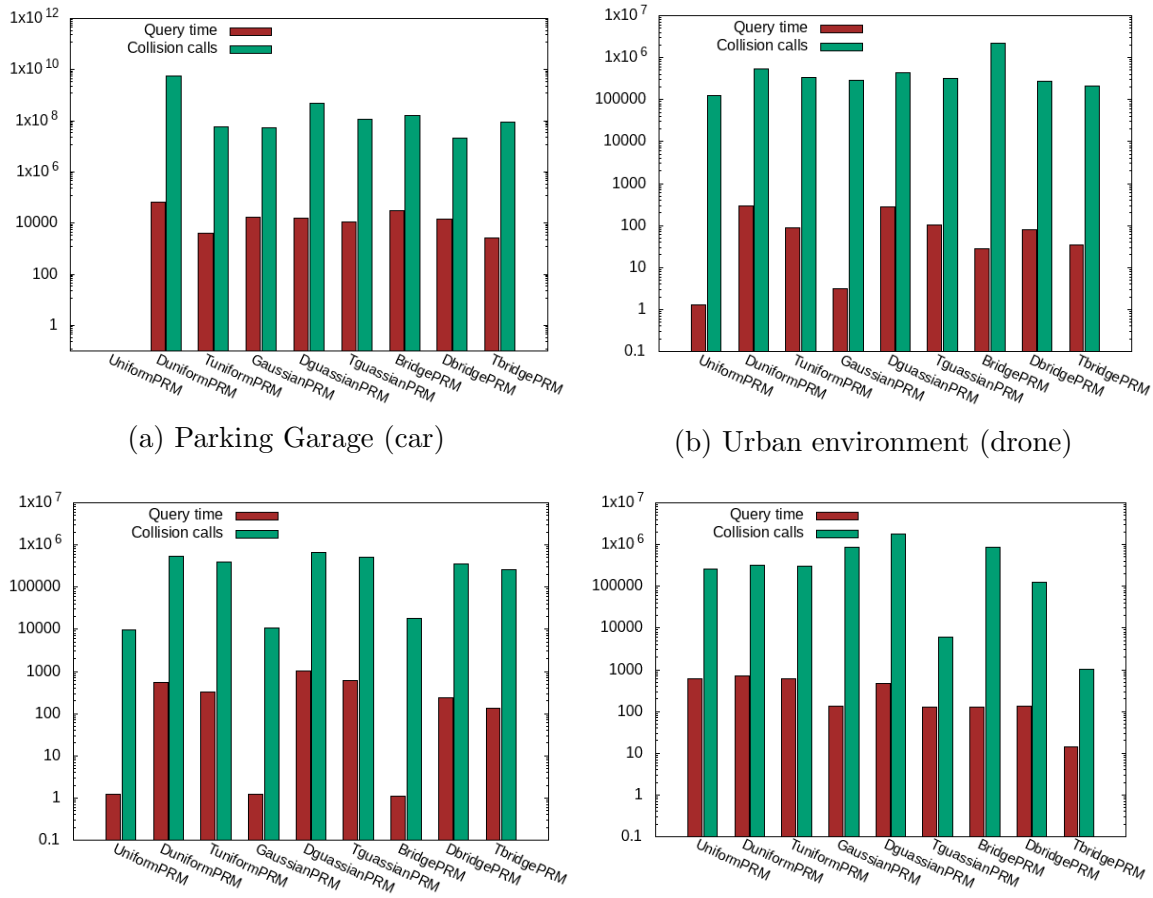
(b) Urban environment (drone)

(c) Kuka YouBot environment

(d) PR2 robot environment

Figure 3.17: The figure shows the total number of nodes present in the roadmap and the size of the largest connected component (CC) for all environments. The x-axis labels with the prefix 'D' indicate density map methods, and the prefix 'T' denotes the topology map methods for PRM planners.

Query time and collision calls: Figure 3.18 shows that PRM took less time to generate a path than density map-based PRM and topology map-based PRM in Urban and Kuka YouBot environments for all three samplers. This difference is due to the additional pre-processing time needed for topology or density map generation. Topology map-based methods achieved fewer collision calls with the shortest query time in complex environments such as Parking Garage (maze-like structure) and PR2 robot environments (high DOF). As a result, we can conclude that the computation time overhead for generating a topology map becomes negligible for our approach when dealing with complex environments.



(a) Parking Garage (car)

(b) Urban environment (drone)

(c) Kuka YouBot environment

(d) PR2 robot environment

Figure 3.18: The figure shows the total query time and the number of collision calls invoked for all PRM methods.

Path Quality: As inferred from Table 3.8, topology map-based PRM methods produced shorter paths than compared methods. It also demonstrates that the preserved important nodes from the density map after the simplicial collapse and the feasible critical points information in the topology map were crucial in bringing edges closer to $\mathcal{C}_{obst}$. As a result, the topology map's path cost converges to a near-optimal value much faster.

3.2.4.4 Comparison with state-of-art methods in a 2D environment

In this environment, we compare the performance of the different planners, i.e., RRT, RRT*, PRM, and PRM*, using our pre-processed maps with the results from [66] to show the improvement in path quality as the sampling density increases. The methods like RRT* and PRM* are proven to converge to an optimal solution as the number of samples increases.

Table 3.8: Path cost achieved by different PRM planners

PRM Planners	Parking Garage	Urban	Kuka YouBot	PR2 robot
Uniform	DNF	2552.5	105.2	10
Uniform with density map	12027	2495.33	92	12
Uniform with topology map	11450.6	2420.1	**85**	4
Gaussian	14676.7	2493.4	104.6	12
Gaussian with density map	11668.75	2278.78	89	15
Gaussian with topology map	11540	**2139.8**	**85**	5
Bridge-Test	14252	2773.4	116.1	11
Bridge-Test with density map	11812	2377.33	92	16
Bridge-Test with topology map	**11307.5**	2377.4	89	**3**

We record the behavior of these methods for our topology map to understand the difference in the result from [66].

We performed experiments for sampling densities of 500, 1000, 2500, 5000, 10000, and 15000. We did not sample beyond 15000 nodes because the Hausdorff distance does not change after reaching the lowest constant value, the reason explained in [126]. We analyzed the behavior using the uniform sampling strategy density map and topology map.

Path Cost and Time: In Figure 3.19a, we observe that as the number of samples increase, the path cost decrease and reaches a minimum cost value for the density map. On the other hand, the methods were able to show a similar pattern, i.e., a gradual decline in path cost, using our topology map, and, as the sampling condition gets fulfilled at the last sampling density, the path cost attains a minimum value at an earlier stage. In Figure 3.19c, we observe that the methods take less node connection time and fewer collision calls using our topology map than the density map. We notice a similar trend for PRM and PRM* methods in Figures 3.19b and 3.19d. Thus, the methods showed an early convergence to the optimal solution using our topology map.

Path Clearance vs Path Cost: Recall that path clearance defines the offset distance of the path from the obstacle boundary. Therefore, as the number of nodes increases, more nodes get identified closer to the objects leading to a decrease in path cost and pathway clearance. As expected in Figure 3.19e, the path clearance decreases with a decrease in path cost using both maps. However, we see a consistency in the pattern as the values decline using

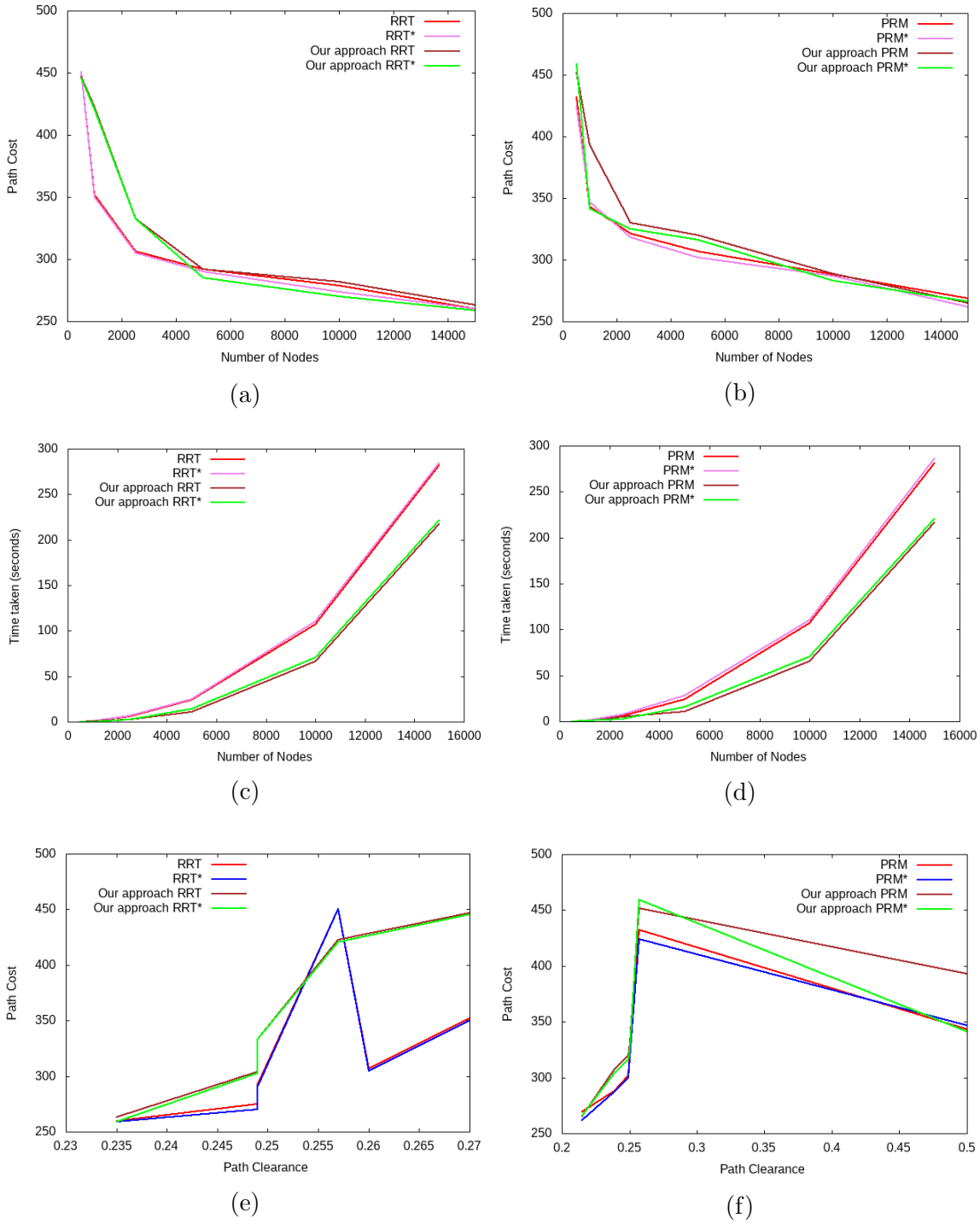


Figure 3.19: Plots showing performance of RRT, RRT*, PRM, and PRM* in the 2D environment compared with our approach for Path cost vs. Number of Nodes in (a) and (b), Time taken vs. Number of Nodes in (c) and (d), and Path Cost vs. Clearance in (e) and (f).

the topology map compared to the sinusoidal pattern observed using the density map. We analyze a similar trend in Figure 3.19f. Overall, we conclude that while using our topology

map, our methods achieve better paths with near-optimal properties at a faster convergence rate compared to results from [66].

Hence, we observed that geometric features added to our previously constructed roadmap in Section 3.1 improve the quality of paths, thus, showing near-optimal characteristics. The roadmap presented the compact representation of the $\mathcal{C}_{space}$ by capturing only the intricate details of the space and holds potential for use in coverage path planning, diverse path planning, etc.

3.3 Incremental Path Planning

This work [123] combines the topological and geometric features extracted in Sections 3.1 and 3.2 for incremental path planning in a partially observable space. A partially observable environment refers to the robot's perspective that the robot observes at each time step. At every time step, the robot can see only a part of the environment that is reachable in its field of view. The work contributes a novel algorithm that incrementally glues the simplices of partially observable space while planning routes in the configuration space. We denote each visible sub-region of the environment as a voxel polytope. The algorithm provides a sub-optimal solution by generating a near-optimal local path in each voxel and gluing them together to find a global route from the start to the goal positions of the robot. In this way, it incrementally applies the topological data analysis approach to each voxel block by preserving only necessary configurations and edges representing the unique property of the subspace. Figure 3.20 shows a pictorial representation of different steps of our proposed algorithm.

3.3.1 Preliminaries

Our incremental topology path planner uses the concept of voxel polyhedron [103] to consider partially observable $\mathcal{C}_{space}$ into sub-regions and perform metric gluing to explore and cover the underlying planning space.

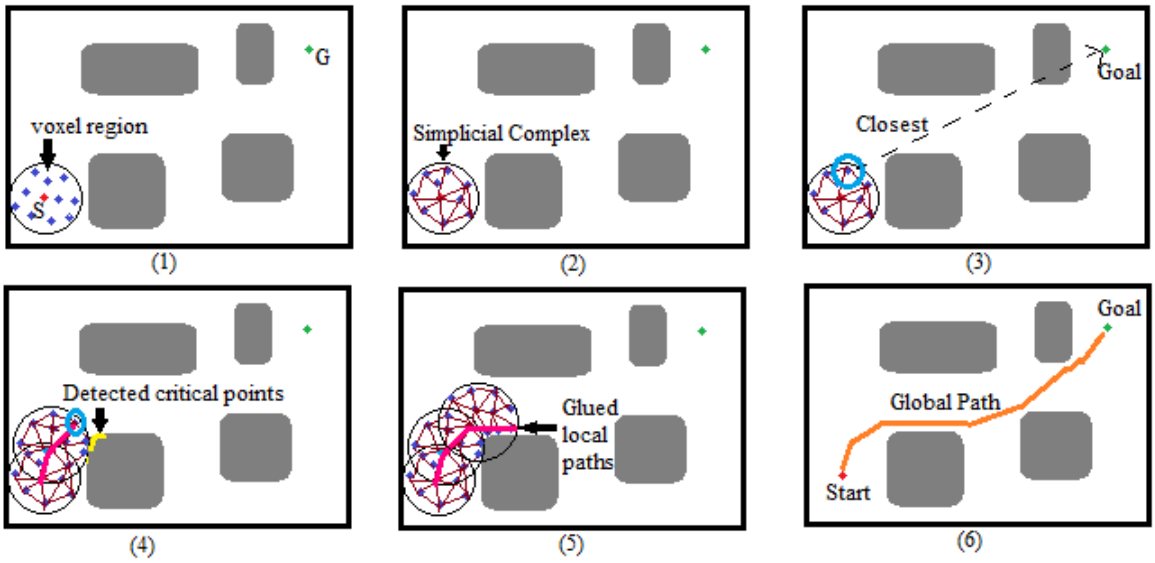


Figure 3.20: A graph illustration of our algorithm in $\mathbb{R}^2$ space where S represents the start point of the robot and G refers to the goal point.

3.3.1.1 Voxel Polytope

Different from existing incremental work on roadmap construction, whose incremental block is in the form of samples, our proposed incremental approach is in batches, and our blocks are in the form of simplices. The batched step-by-step process generates local roadmaps of the sub-space of $\mathcal{C}_{free}$ decomposed as voxel polytope. The increase in the number of voxels glued together is proportional to the increase in the $\mathcal{C}_{free}$ area covered by the robot resulting in a higher probability of finding a solution if one exists. For each voxel block/window, we consider the spherical boundary of radius λ given as

$$\lambda = n * d, \tag{3.8}$$

where n is the dimensionality of $\mathcal{C}_{space}$ and d is the diameter of the circumscribed circle of the robot. So, the length of the global path becomes equal to the total number of glued voxel blocks times λ, i.e., gluing the local paths from each connecting voxel. The connection edge between the configuration nodes in each voxel window forms at a connection length γ. We compute the value of γ as

$$\gamma = min(\log(|V_i|)/|V_i|, \varepsilon); i \geq 0, \tag{3.9}$$

where $|V|$ is the number of the vertices generated in each voxel window i to meet the sampling condition from [126]. ε is the constant value provided by the user. Here, the boundary of each voxel window gets considered as the criteria for satisfying the sampling condition. The generated simplices in each voxel block undergo a simplicial collapse to obtain a 1-dimensional skeleton of simplices. Our density-based discrete Morse function, from [124], applied on the skeleton structure identifies critical points and feasible critical points in proximity of $\mathcal{C}_{obst}$. As a result, our method stores the topological and geometric information in the local topology map of respective voxels. Next section, we discuss the theoretical proof of metric gluing our local topology maps together.

3.3.1.2 Metric gluing of simplicial complexes

The work [4] proved that the Vietoris-Rips complex of two metric graphs glued together along a sufficiently small sub-complex is homotopy equivalent to the union of the Vietoris-Rips complexes.

In general, the term metric space refers to a set X equipped with a distance function $m\colon X^2 \to X$ satisfying the conventional axioms: non-negativity, symmetry, the triangle inequality, and the property that the value 0 is achieved exactly on the pairs of the type (x, x). Let us assume two metric spaces X_1 and X_2 with a set-theoretic intersection $X_1 \bigcap X_2$. We say the intersection on metric space is well-defined if the two distance functions m_1 and m_2 agree on $(X_1 \bigcap X_2)^2$. Notice that in general, even in this case, the distance function on the set-theoretic union $X_1 \cup X_2$ is not well-defined in a natural way, and so, there can be more than one metric on $X_1 \cup X_2$ that restricts to the given metrics on X_1 and X_2, but if X_1 and X_2 are subsets of a larger metric space X and the metrics are the restrictions of an ambient metric m on X then we can conclude that m gives a unique natural extension of both m_1 and m_2 to the restriction of m on $X_1 \bigcup X_2$. When we take another perspective where m_1 and m_2 are path metrics as usually defined in graphs, the path metric on the union of graphs is, in fact, a well-defined extension. Accordingly, our metric spaces are either sub-regions of $\mathcal{C}_{free}$ or sub-graphs of the ambient graph obtained as the 1-dimensional skeleton of the Vietoris-Rips complex from samples on $\mathcal{C}_{free}$.

A $\mathcal{C}_{space}$ is a topological space, so we apply the properties of metric gluing in it to join the simplices of our voxel graphs. Suppose D_A and D_B represent the simplicial complex

obtained after the collapse on metric spaces A and B, and H is a sub-complex on the vertices common to both A and B metric spaces. The function *diam* refers to the circle diameter that circumscribes the metric space. For a subset Q of a metric space, we define the diameter $diam(Q)$ as the supremum of values $m(x, y)$ over all choices of x, $y \in Q$. Theorem 8, as stated in [4] by Adamaszek et al., reads as follows.

Theorem 3.5 *Let A and B be metric spaces with $A \bigcap B = H$, where H is a closed subspace of A and B, and let $r > 0$. Consider $A \bigcup_H B$, the metric gluing of A and B along the intersection H. Suppose that if $diam(D_A \bigcup D_B) \leq r$ for some $\Phi \neq D_A \subseteq A \backslash H$ and $\Phi \neq D_B \subseteq B \backslash H$, then there is a unique maximal nonempty subset $\sigma \subseteq H$ such that $diam(D_A \bigcup D_B \bigcup \sigma) \leq r$. Then $R(A \bigcup_H B; r) \simeq R(A; r) \bigcup_{R(H;r)} R(B; r)$. Hence if $R(H; r)$ is contractible, then $R(A \bigcup_H B; r) \simeq R(A; r) \vee R(B; r)$.*

The theorem proves that gluing Vietoris-Rips complexes of finite metric spaces A and B provides a simplex homotopy equivalent to the Vietoris-Rips complex of the $A \bigcup B$ metric space glued along the closed subspace H. A conclusion from Corollary 9 in [4] is as follows.

Corollary 3.1 *Let A and B be metric spaces with $A \bigcap B = H$, where H is a closed subspace of A and B, and $A \bigcup_H B$ is their metric gluing along H. Let $r > 0$, and suppose $diam(H) \leq r$. Then $R(A \bigcup_H B; r) \simeq R(A; r) \vee R(B; r)$.*

Further, the conclusion from Theorem 3.5 extends to show its use in metric graphs, from Theorem 10 in [4], reads as follows.

Theorem 3.6 *Let $G = G_A \bigcup_{G_H} G_B$ be a metric graph, where $G_H = G_A \bigcap G_B$ is a closed metric subgraph of the metric graphs G_A and G_B. Suppose furthermore that G_H is a path and that each vertex of G_H besides the two endpoints has degree 2, not only as a vertex in G_H but also as a vertex in G. Suppose the length of G_H is at most $l/3$, where any simple loop in G that goes through G_H has a length of at least l. Let $A \subseteq G_A$ and $B \subseteq G_B$ be arbitrary subsets such that $A \subseteq G_A = B \subseteq G_B = A \bigcap B := H$. Then $R(A \bigcup_H B; r) \simeq R(A; r) \bigcup_{R(H;r)} R(B; r)$ for all $r > 0$. Hence if $R(H; r)$ is contractible, then $R(A \bigcup_H B; r) \simeq R(A; r) \vee R(B; r)$.*

We extend this theory to high-dimensional spaces with paths greater than degree 2, i.e., in $\mathcal{C}_{space}$. So, our proposition states as follows.

Proposition 3.1 *Let G_A and G_B be metric/voxel graphs with $G_A \bigcap G_B = G_H$, where G_H is a closed sub-graph of G_A and G_B with vertices of degree greater than 2, and $G_v = G_A \bigcup_{G_H} G_B$ is the metric gluing of two metric graphs along G_H. Let $r \geq 2 * \lambda > 0$, the voxel radius from Eq. 3.8, and suppose the $diam(G_H) \leq r$. Then $R(A \bigcup_H B; r) \simeq R(A; r) \vee R(B; r)$.*

Proof: Let G_v, G_A, G_B, G_H, A, B, and H satisfy the same hypotheses as in the statement of Theorem 3.6. Let z be a vertex in G_A that extends to the voxel graph G_B, such that $z \in G_A \bigcap G_B$ and voxel B is centered at z. We claim that no point $p \in G_v \setminus G_H$ is within distance r of G_H. Indeed, if there were such a point $p \in G_v \setminus G_H$ satisfying $d(p, u) \leq r$ and $d(p, v) \leq r$ where d is the Euclidean distance function and $u, v \in G_H$, then we could produce a homotopically non-trivial loop through the gluing simplex in G_H with local path shorter than λ, giving contradiction to our assumption. It follows that if the maximal non-empty set of G_A and G_B has $diam(\sigma_A \bigcap \sigma_B) \leq r$, then there exists a unique maximal clique σ_H connecting the simplicial complexes $R(A)$ and $R(B)$. From Theorem 3.6, we can conclude that $R(A \bigcup_H B; r) \simeq R(A; r) \vee R(B; r)$ can extend for configurations of degree greater than 2. Figure 3.21 shows an example of gluing two voxel complexes. □

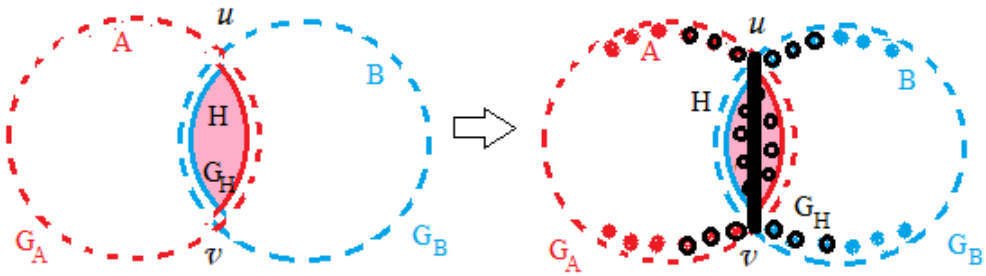


Figure 3.21: Illustration of voxel graph gluing for G_A and G_B

Our work utilizes the metric gluing property of the Rips complex to connect local paths contributing towards a global pathway formation from the start to the goal position. The incremental approach first constructs the local Rips complex that is topologically equivalent to a sampled sub-space and then glues different pieces together to cover larger sections of the sampled $\mathcal{C}_{free}$ space.

3.3.2 Approach to Incremental Path Planning

Algorithm 3 creates voxel polytopes by taking an extending configuration as its center until the robot reaches/connects the goal configuration. The bounding radius of the voxel gets computed from Eq. 3.8. At each voxel, the algorithm generates a topology map of the local region from steps in [126, 124] and plans the local path from start to an intermediate configuration, and so forth until it connects to the goal configuration. It constructs the Rips complex using method *ConstructComplex* and performs simplicial collapse to obtain a 1-dimensional skeleton of simplex using *TopologicalCollapse* method. The density-based discrete Morse function applied on each local topological map extracts critical points information and the configurations (or feasible critical points) at proximity to $\mathcal{C}_{obst}$ using methods *IdentifyCriticalPoints* and *GetValidConfigurations*, respectively. In line 9, the algorithm considers the vertices of the simplicial complex and feasible critical points to plan a memory-efficient route at a ϱ-clearance from obstacles in method *PlanPath*.

It performs Breadth-First Search (BFS) to select a suitable configuration closest to the goal compared to all configuration nodes from the ambient voxel graph, i.e., a goal-biased expansion, in lines 11-13. It ensures that the selected configuration node for extension is neither repeated nor rejected node evaluated previously. The algorithm selects a random node in the convex hull of S of the immediate voxel polytope in lines 14-16.

The algorithm extends voxels by considering nodes from the convex hull of S of the immediate voxel polytope. It narrows down its selection to only nodes in the semicircle of the convex hull facing or closest to the goal position in line 11. It randomly selects a node from N for expansion until it succeeds and discards failed configurations to avoid redundancy, lines 12-17. It keeps performing these steps for each extended voxel and incrementally glues simplices or the planned local path of the voxel until the query is solved or it reaches a computation limit. As a result, the algorithm returns a global pathway P connecting the start and goal configurations and the roadmap M.

3.3.3 Experimental Setup

We perform experiments in seven different environments with robots ranging from 3 DOF to 14 DOF, as discussed next.

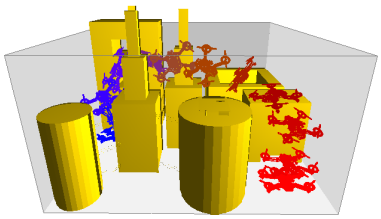
Algorithm 3 Incremental Path Planner

Input: G: A dense graph comprising of a vertex set V and edge set E where $G = \{V, E\}$, Q: A query to be solved from a start to a goal position, S: Simplicial Complex, Z: Voxel Polytope, λ: Radius of voxel polytope, p: Local path set, P: Global path set, M: Glued voxel graphs, N: Nodes set.

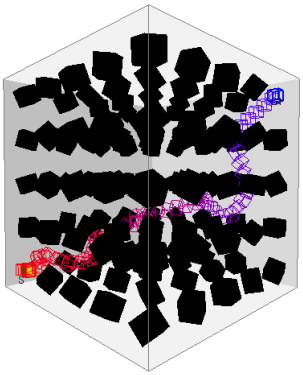
Let $P \leftarrow null$, $p \leftarrow null$, q_0 = start configuration.
Create a voxel polytope Z(q_0, λ).
while Q not solved **do**
  if Boundary(Z) not in collision with the robot **then**
    $S \leftarrow ConstructComplex(G, Z)$; ◁ Refer Sec. 3.1
    $TopologicalCollapse(S)$; ◁ Refer Sec. 3.1
    $C \leftarrow IdentifyCriticalPoints(S)$; ◁ Refer Sec. 3.2
    $F \leftarrow GetValidConfigurations(S, C)$; ◁ Refer Sec. 3.2
    p = PlanPath(Z, S, F)
    if goal $\notin$ P **then**
      $N = ClosestToGoal(ConvexHull(S))$
      while N is not empty **do**
        $q = SelectRandomNode(N)$
        if q expands Z and $q \neq q_0$ **then**
          Create a voxel polytope $Z_q(q, \lambda)$.
        else
          $N = N \setminus q$
      Z $\leftarrow Z_q$
      $q_0 = q$
      $Z_q \leftarrow null$
  else
    $\lambda = 2 * \lambda$
    Create a voxel polytope Z(q_0, λ).
  $P = P \bigcup p$
  $M = M \bigcup S$
return $\{P, M\}$

- **Parking Garage**: The 3D environment has a vehicle parking garage structure, as shown in Figure 3.22a. The robot is a planar car with 3 DOF.
- **Urban environment**: This environment consists of buildings as obstacles in the city-like structure, as shown in Figure 3.22b. The robot is a 6 DOF drone.
- **Cluttered environment**: Obstacles are cluttered around the room as shown in Figure 3.22c. The robot has to traverse through these obstacles successfully to reach its goal. The robot is a 6 DOF cube.

(a) Parking Garage

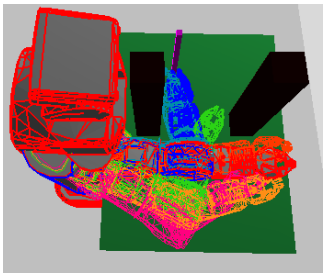


(b) Urban environment

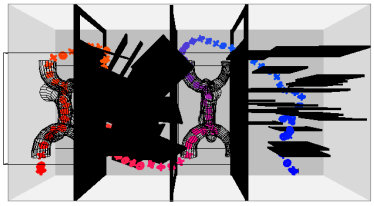


(c) Cluttered environment

(d) Kuka YouBot environment



(e) PR2 robot environment



(f) Heterogeneous 3D

(g) 3D Tunnel

Figure 3.22: Environments Studied

- **Kuka YouBot environment**: An 8 DOF robot in an environment with four different rooms, see Figure 3.22d. The robot moves through these rooms within narrow passages and arrives at its destination, where it performs an action (grasps or puts an object down). This robot is a simulation replica of Kuka YouBot [72].

- **PR2 robot environment**: The environment has two pillars and a table as the obstacles where the robot is required to bypass through the pillars to grasp the stick kept on the other side, as shown in Figure 3.22e. The robot is a simulation replica of the PR2 robot [140] with only the right-hand arm (14 DOF) and fixed base.

- **Heterogeneous 3D**: A 3D maze environment with walls and narrow passages between the walls. A robot has to pass through maze-like tunnels to reach the goal, as shown in Figure 3.22f. The robot is a 6 DOF toroidal plus.

- **3D Tunnel**: This environment comprises a narrow passage tunnel and a set of cluttered cubes, as shown in Figure 3.22g. The robot is a 9 DOF articulated linkage chain that has to pass through a hole in the wall and encounter obstacles at the other end of the wall.

3.3.4 Result Analysis

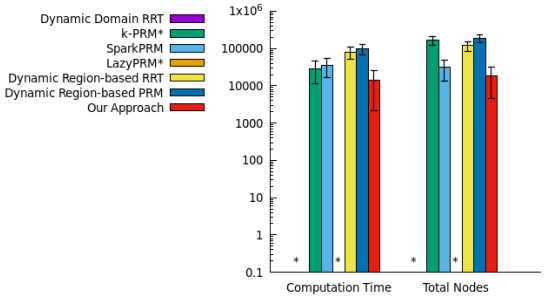
We compare our proposed method with optimal planners k-PRM^* [67] and LazyPRM^* [61], with cell decomposition methods Dynamic Domain RRT [147] and Dynamic Region-based RRT/PRM [44, 105], and with Sampling-based RoadMap Trees (SRT) planner SparkPRM [109] which combines a PRM and RRT approach. All results were averaged over ten randomized seeds with an evaluation of 500 trials. The value of $\varepsilon = 1$ in Eq. 3.9. We used the PRM strategy to uniformly sample configurations within each voxel.

3.3.4.1 Optimal Planners

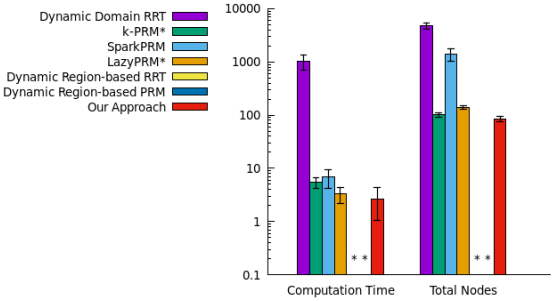
We consider the value of k = 15 for k-PRM^* as derived in Karaman et al. paper on optimal path planning [67] and the connection radius as the function of random configuration nodes in $\mathcal{C}_{space}$ for both k-PRM^* and LazyPRM^* methods.

Computation Time and Total Nodes: We plot our results considering the time and total nodes needed to solve the query as seen in Figure 3.23 and report that our method performs faster than the optimal planners and needs fewer nodes in all the environments except the Cluttered environment (Figure 3.23c). k-PRM^* achieves the lowest computation time, and LazyPRM^* requires the least number of nodes in the roadmap. The minimal overhead of our planner in the Cluttered environment is due to the time consumed to process topological and geometric information at each voxel. As our results have shown, our planner improves as the DOF of the robot increases, and the Cluttered environment comprises a simple box robot in a uniformly distributed space.

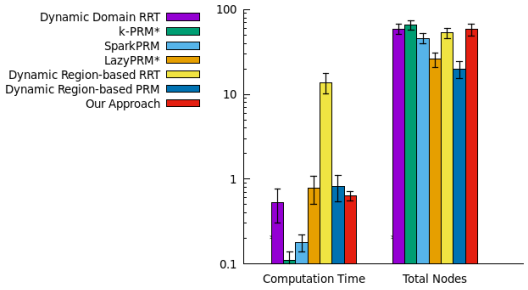
Average Path Cost: Table 3.9 shows all methods' average path cost with standard deviation (superscript) values. Compared to the extra processing time used by our planner, it still produced the shortest path in all environments. LazyPRM^* failed to finish finding a



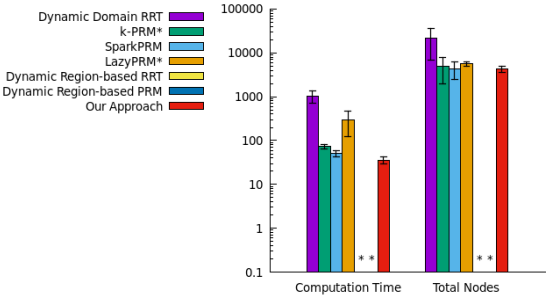
(a) Parking Garage (3 DOF car)



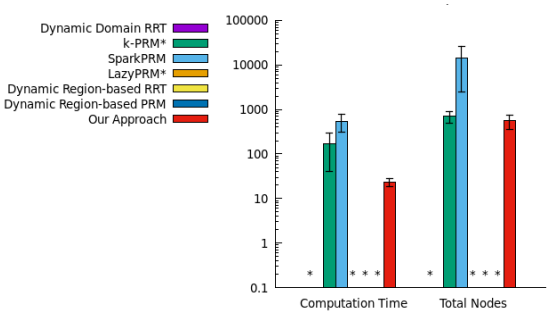
(b) Urban environment (6 DOF drone)



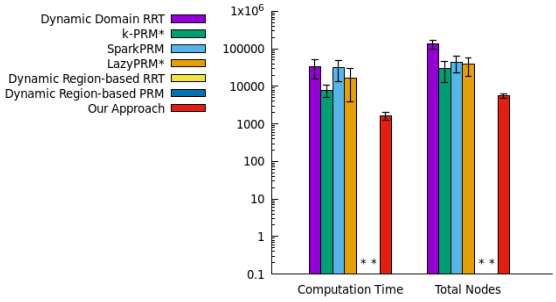
(c) Cluttered environment (6 DOF cube)



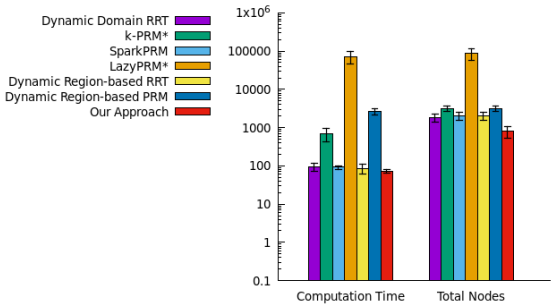
(d) Kuka YouBot environment (8 DOF)



(e) PR2 robot environment (14 DOF)



(f) Heterogeneous 3D (6 DOF toroidal plus)



(g) 3D Tunnel (9 DOF serial chain)

Figure 3.23: The plots for each environment show the total computation time (in seconds) and the number of nodes in a roadmap recorded for each planner averaged over ten random runs. The error bars show the standard deviation. "*" indicates no result data available for the respective planner or the planner failed to finish within 72 hrs.

path in the Parking Garage and PR2 robot environments within 72 hrs. Since LazyPRM^* continually increases the density of its approximation, the graph search eventually becomes too expensive and fails to find solutions in complex environments. The abbreviation DNF stands for "Did Not Finish."

Table 3.9: Path Cost computed for Optimal Planners

Environments	**Our Approach**	**k-PRM***	**LazyPRM***
Parking Garage	**11139.5**$^{\pm 105.5}$	11879$^{\pm 118.9}$	DNF
Cluttered environment	**402**$^{\pm 12.1}$	425$^{\pm 26.1}$	463$^{\pm 17.4}$
Urban environment	**1750**$^{\pm 4.8}$	1823$^{\pm 36.2}$	1758$^{\pm 14.8}$
Heterogeneous 3D	**2341.4**$^{\pm 18.5}$	3640$^{\pm 19.3}$	3611$^{\pm 9.6}$
3D Tunnel	**293.3**$^{\pm 3.9}$	337$^{\pm 6.5}$	407$^{\pm 17.2}$
Kuka YouBot environment	**3601.3**$^{\pm 24.5}$	9405$^{\pm 46.7}$	9211$^{\pm 20.9}$
PR2 robot environment	**10**$^{\pm 0.8}$	11$^{\pm 1.3}$	DNF

3.3.4.2 Cell Decomposition Methods

We compare our method with two different cell decomposition strategies that also analyze the properties of $\mathcal{C}_{space}$. We combined the dynamic region sampling strategy from [44, 105] with RRT and PRM methods and performed experiments in all seven environments.

Computation Time and Total Nodes: Our method extracts the topological and geometric representation of the $\mathcal{C}_{space}$ at each voxel and metrically glues them to maintain the topological connectivity between the voxel graphs. We can deduce from Figure 3.23 that the overhead of approximating the $\mathcal{C}_{free}$ information becomes negligible as the complexity of $\mathcal{C}_{space}$ increases. Hence, we observe a boost in the performance of our planner with the lowest computation time and least number of nodes in six of our seven environments studied. Via Figure 3.23c, we report that Dynamic Domain RRT uses less time than other cell decomposition methods, whereas Dynamic Region-based PRM requires fewer nodes to plan a path in the Cluttered environment. However, Dynamic Region-based RRT/PRM could not succeed in solving queries for the Urban environment, Heterogeneous 3D, and Kuka YouBot environment. All three baseline methods failed to search a route in the PR2 robot environment, and Dynamic Domain RRT could not finish finding pathways in the Parking Garage within 72 hrs.

Average Path Cost: Table 3.10 shows the average path cost attained by the cell decomposition methods. We observe that using the topological and geometric information preserved at each voxel does not affect the homotopy-equivalence representation of our roadmaps, and our approach was still able to obtain edges closer to the $\mathcal{C}_{obst}$ generating the shortest path in six of our seven environments compared to other baseline methods. Dynamic Region-based PRM finds the shortest route compared to other planners in the Cluttered environment. The acronym used in our table for Dynamic Region-based RRT and PRM is DRb-RRT and DRb-PRM.

Table 3.10: Path Cost computed for Cell Decomposition methods

Environments	**Our Approach**	**Dynamic Domain RRT**	**DRb-RRT**	**DRb-PRM**
Parking Garage	$\mathbf{11139.5}^{\pm 105.5}$	DNF	$13647^{\pm 19.1}$	$16982^{\pm 130.3}$
Cluttered environment	$402^{\pm 12.1}$	$419.89^{\pm 4.5}$	$433.66^{\pm 5.8}$	$\mathbf{364}^{\pm 16.1}$
Urban environment	$\mathbf{1750}^{\pm 4.8}$	$1761.5^{\pm 12.9}$	N/A	N/A
Heterogeneous 3D	$\mathbf{2341.4}^{\pm 18.5}$	$3483^{\pm 22}$	N/A	N/A
3D Tunnel	$\mathbf{293.3}^{\pm 3.9}$	$336.04^{\pm 6}$	$337.69^{\pm 6.1}$	$396^{\pm 7.9}$
Kuka YouBot environment	$\mathbf{3601.3}^{\pm 24.5}$	$3623^{\pm 25}$	N/A	N/A
PR2 robot environment	$\mathbf{10}^{\pm 0.8}$	N/A	N/A	N/A

The dynamic region-based sampling strategy performs tetrahedral workspace decomposition of the $\mathcal{C}_{free}$ and removes the $\mathcal{C}_{obst}$ at each process, representing them as holes. Since this strategy incrementally decomposes $\mathcal{C}_{free}$ until it solves the query, the presence of $\mathcal{C}_{obst}$ in a non-uniform fashion within tight gaps in the $\mathcal{C}_{space}$ eliminates the crucial connecting regions of $\mathcal{C}_{free}$ while constructing the Reeb graph, which results in its failure to find a solution in environments with non-uniformity.

As explained in Section 3.3.2, our method performs validity checks, convex hull computations within sub-regions, and the distance metric test to find the closest configuration to the goal for the next voxel polytope. Overall our method avoids heavy geometric evaluation that requires Voronoi cell computation, sweep-line validity check, or pruning objects into holes. These computational methods start degrading in their performance when dealing with high DOF robots or cluttered/non-uniform environments. Hence, our planner becomes more successful than the baseline methods.

3.3.4.3 Sampling-based RoadMap Trees (SRT) Planners

We compare our approach with SparkPRM as it combines the capability of PRM and RRT methods to find a solution in an environment similar to ours.

Computation Time and Total Nodes: From Figure 3.23g, we observe a slight difference in computation time between SparkPRM and our method, which indicates an improvement in the efficiency of our planner as it extracts the features of $\mathcal{C}_{space}$. The performance of our method improves as the complexity of the $\mathcal{C}_{space}$ increases with importance placed on the preserved topology information, which influenced the generation of configurations in proximity to $\mathcal{C}_{obst}$, thus creating a better path.

Our method uses less time and fewer nodes to solve the query than SparkPRM in six of the seven environments. In the Cluttered environment, SparkPRM performs better than our approach in computation time and generates fewer nodes to solve a query, as shown in Figure 3.23c. It indicates that the topological and geometric extraction performed along with metric gluing by our method becomes a little time-consuming here.

Average Path Cost: Although SparkPRM finds the shortest path in the Cluttered environment, our method attains convergence to low path cost in the remaining six testbeds due to the preserved topological and geometric information of the $\mathcal{C}_{space}$ that influences the formation of edges closer to the $\mathcal{C}_{obst}$, in Table 3.11.

Table 3.11: Path Cost computed for SRT planners

Environments	Our Approach	SparkPRM
Parking Garage	$\mathbf{11139.5}^{\pm 105.5}$	$14915^{\pm 138.6}$
Cluttered environment	$402^{\pm 12.1}$	$\mathbf{336}^{\pm 8.3}$
Urban environment	$\mathbf{1750}^{\pm 4.8}$	$1964^{\pm 30.8}$
Heterogeneous 3D	$\mathbf{2341.4}^{\pm 18.5}$	$2486^{\pm 24.9}$
3D Tunnel	$\mathbf{293.3}^{\pm 3.9}$	$331^{\pm 5.6}$
Kuka YouBot environment	$\mathbf{3601.3}^{\pm 24.5}$	$4648^{\pm 34.8}$
PR2 robot environment	$\mathbf{10}^{\pm 0.8}$	$17.9^{\pm 1.1}$

Based on our analysis, we conclude that our method outperforms the tested baseline methods. Our approach has successfully identified the optimal route from the start to the goal position by metric gluing the voxel blocks while retaining the topological significance.

Hence, our findings indicate that our approach is a promising solution for planning in high-dimensional complex environments.

In conclusion, we introduced three novel approaches for extracting topological and geometric information about the environment. These methods involve pre-processing steps before path planning and incremental extraction while concurrently planning the path. The first part utilizes the Rips complex to capture topological information and discrete Morse theory to capture geometric properties. The second part incrementally obtains topological and geometric information while planning routes in local regions. Our proposed methods have demonstrated substantial enhancements in the performance of sampling-based motion planners and have generated a reusable map of the environment, thus, contributing to the advancement of motion planning techniques. Chapter 4 will discuss the advantages of the topology roadmap in solving some compelling motion planning problems.

CHAPTER 4

A Topology Perspective on Motion Planning Problem

Path planning is an important function needed to successfully move autonomous robots while satisfying numerous constraints, such as avoiding collision with obstacles or bumping into walls in a narrow passage. This chapter discusses our topology-based solution to some compelling problems in robot motion planning. We apply our algorithmic framework to construct the $\mathcal{C}_{space}$ roadmap and use topological and geometric information to find multiple routes to overcome path planning failures in a changing environment [122, 121].

4.1 Diverse Path Planning

While path diversity is already a fruitful topic in robotics, the research in this work points to methods for solutions to a more compelling problem of finding diverse paths that represent distinct homotopy classes of pathways. Suppose the $\mathcal{C}_{free}$ is two-dimensional and not contractible. In this case, let us assume that there is at least one obstacle such that it is homomorphically a disk, as shown in Figure 4.1. Non-homotopic path construction is possible by successively selecting feasible critical points as goalposts near critical points on the boundary of that same disk. Suppose one obstacle in a 2D environment is a free-floating regular polygon (i.e., not touching a wall). We posit that an alternate selection of feasible critical points on diametrically opposite sides of the boundary circle will result in the generation of two paths that are not homotopic to each other. One will "touch" the disk on one side, e.g., in the tangential clockwise direction along the boundary circle (path P in Figure 4.1). The other is near the diametrically opposite point on the other side in the tangential counterclockwise direction (path P$'$ in Figure 4.1). In the coming discussion, we will get into details of the practical application of this idea for identifying coarsely-diverse routes in the configuration space.

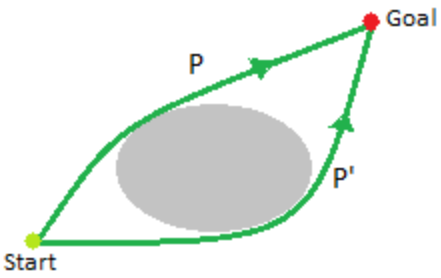


Figure 4.1: The figure shows two coarsely diverse paths, P and P′, incident on either side of the disk. Hence, we say P and P′ represent distinct homotopy classes of trajectories.

4.1.1 Coarsely-diverse Paths

We extend the mathematical concepts discussed in Sections 3.1 and 3.2 to define path diversity formulation for high-dimensional planning space.

Definition 4.1 *(**Critical points**): The set of critical points is defined as the set of non-degenerate points on the convex hull of $\mathcal{C}_{obst}$ when the given discrete Morse function f reaches its extreme values, i.e., local minima or maxima.*

Here, we denote the vertices of Vietoris-Rips complex by S and critical points set as C. The derived feasible critical points information near $\mathcal{C}_{obst}$ was obtained within $\mathcal{C}_{free}$ by essentially shifting the critical points radially at a distance ϱ from $\mathcal{C}_{obst}$ to within S, thus providing a set of vertices in $S \subseteq \mathcal{C}_{free}$. The computation of ϱ value does not affect the identification of feasible critical points and, thus, can be of any choice. Here, we calculate the value of ϱ as defined in [124].

Definition 4.2 *(**Feasible critical points**): This set is defined as all vertices in S at a radial clearance of ϱ from a critical point of $\mathcal{C}_{obst}$. In other words, it is the union of intersections of vertices in S within the metric balls of radius ϱ centered at some critical point.*

Let c_i denote an element of the set C, i.e., $c_i \in C$, and let v_j denote an element of the set S, $\forall i, j > 0$. We denote the list of the feasible critical points within ϱ from c_i by $F(c_i) = \{v_0, ..., v_q\}$, $\forall q, 0 \leq q < size(S)$.

While planning the path in the $\mathcal{C}_{space}$, we consider the critical point information to find the shortest route with minimum clearance to the $\mathcal{C}_{obst}$. So, the planned path is incident on the feasible critical points representing a different group of critical points in F.

To this end, let us consider F as a function from critical points C to the power set of S, i.e., the set of its subsets. If all $F(c_i)$ are disjoint, which happens when all critical points are at least 2ϱ apart, then there is a well-defined function F^{-1} from feasible critical points $F(C)$ back to C. Even when this is not true, we can think of $F^{-1}(v)$ as the pre-image of v under the multi-valued function F. For each path p, we then have $F^{-1}(p)$ that is the union $\bigcup_{v \in p} F^{-1}(v)$ but can also be thought of as a sequence in C because it inherits the order from p. Suppose d is the Hausdorff distance function that measures the distance between two paths at time t. We define the path diversity as below.

Definition 4.3 *(**Path diversity**): Given a set of paths P, two paths p_a and p_b in P are diverse if $p_a \neq p_b$. They are called coarsely diverse, or more precisely ϱ-coarsely diverse if for some time $t > 0$ we have distance $d(p_a(t), p_b(t)) > \varrho$. Two paths that are not coarsely diverse are called fellow travelers.*

The fellow travelers are assumed to be paths of the same homotopy class from Def. 2.1.

Proposition 4.1 *Suppose the distance between any two critical points is at least 3ϱ. Then two paths p_a and p_b in P are coarsely diverse if $F^{-1}(p_a) \neq F^{-1}(p_b)$.*

Proof: The inequality means that for some $t > 0$, $F^{-1}(p_a(t)) \neq F^{-1}(p_b(t))$, so $d(F^{-1}(p_a(t)), F^{-1}(p_b(t))) > 3\varrho$. Since $d(F^{-1}(p_a(t)), p_a(t)) < \varrho$ and similarly $d(F^{-1}(p_b(t)), p_b(t)) < \varrho$, by the triangle inequality $d(p_a(t), p_b(t)) > \varrho$ as needed. □

Proposition 4.2 *The equation $F^{-1}(p_a) = F^{-1}(p_b)$ defines an equivalence relation on paths. All paths in the same equivalence class are pairwise fellow travelers within the bound 2ϱ.*

Proof: The first statement is clear. For the second notice that $d(p_a(t), p_b(t)) \leq 2\varrho$ for all $t > 0$ as $d(p_a(t), F^{-1}(p_a(t)) \leq \varrho$, $d(p_b(t), F^{-1}(p_b(t)) \leq \varrho$, and by the assumption $F^{-1}(p_a(t)) = F^{-1}(p_b(t))$ for all $t > 0$. □

We can conclude that in order to obtain a collection of diverse paths, or even better, coarsely-diverse pathways, it suffices to ensure the collection contains routes that map via F^{-1} to distinct sets of critical points.

Corollary 4.1 *Suppose c is in $F^{-1}(p_a)$ but is not in $F^{-1}(p_b)$. Then p_a and p_b are coarsely diverse.*

4.1.1.1 Algorithmic Development

In this algorithm, multiple paths map to a different collection of critical points incidents to them, as defined in Def.4.3. Critical points represent the important intersection points on the surface of $\mathcal{C}_{obst}$, i.e., high or low curve points. These critical points are non-degenerate and, so, do not change with a change in the dimensionality of the robot, as $\mathcal{C}_{obst}$ are rigid (non-deformable). The number of critical points for a $\mathcal{C}_{obst}$ in 2D space, e.g., a square, will have a small countable value. But a polyhedral representation of $\mathcal{C}_{obst}$ can have enormous critical points due to its intricate structure in 3D space. In such a framework, the possibility of having multiple paths increases around a single $\mathcal{C}_{obst}$ using its critical points information. So, the pruning process removes only those critical points close to the pathway within distance 2ϱ from the identified critical points set. After pruning, there exists a possibility that different paths map to the untouched critical points of the same $\mathcal{C}_{obst}$ in the future. Therefore, it also validates the output paths to ensure they pass along different groups of $\mathcal{C}_{obst}$ using *Proposition 1.*

Algorithm 4 finds coarsely-diverse paths using identified critical points information and pre-computed simplicial complex from our previous work [126, 124], in lines 2-5. The algorithm calls *ConstructComplex* method to construct simplicial complex S (Def.3.1) on a verified Hausdorff distance measure of the dense graph G. It performs a sequence of simplicial collapse on S in the method *TopologicalCollapse*. The discrete Morse function f on S identifies critical points as the computed Morse values in eq.(3.4) reach its extrema in method *IdentifyCriticalPoints*. Finally, it executes *GetFeasiblePoints* method to get vertices in S at ϱ-clearance (Def.4.2) from the critical points of $\mathcal{C}_{obst}$. The resulting topology map is a complete connected graph combined with necessary topological information, i.e., vertices and edges, of the $\mathcal{C}_{free}$ and the critical points information. The algorithm solves a query from start to goal position and uncovers different paths using this roadmap in the $\mathcal{C}_{space}$. At

each iteration (line 6), the algorithm updates the graph while pruning the sets of feasible critical points and constituting critical points of the already found path such that no future pathway passes through the same group set.

As a result, when the local planner begins to build a path for every updated graph complex S, it provides a path incident on feasible critical points representing a different collection of critical simplices compared to all other routes previously obtained (line 8). The algorithm terminates on computing all possible combinations of the paths for the environment such that solving the query further generates no new pathway with the available vertices in the S (line 18).

Once the algorithm finishes computing all the possible paths, it performs a validation to accept only those paths that map via F^{-1} to distinct sets of critical points of coarsely-diverse path classes and lists all the $\mathcal{C}_{obst}$ a particular path covers (lines 20-25). Finally, the algorithm outputs the total number of pathways, i.e., n, and the set of coarsely-diverse paths, i.e., P, for a given $\mathcal{C}_{space}$.

In the analyzed set of paths returned from Algorithm 4 based on the different $\mathcal{C}_{obst}$ groups they cover using critical point information, we noticed many combinations of such paths are possible in a cluttered environment. For such a case, rather than retrieving hundreds of routes for the environment, we limit the output to a threshold to determine the retrieval number of paths. We define the threshold as the range from the shortest path length to a length T, where T is the mean of the shortest path lengths possible in all subregions of the $\mathcal{C}_{free}$. We discuss more details of our result in Section 4.1.3.

4.1.2 Experimental Setup

We performed experiments in three different environments, as described below. Figure 4.2 shows the start and goal positions in red and blue colors, respectively.

- **3D Cluttered environment**: Obstacles are cluttered around the room as shown in Figure 4.2a. The robot has to traverse through these obstacles successfully to reach its goal.
- **House environment**: A L-shaped robot placed in a house-like space with four different rooms, see Figure 4.2c. The obstacles, like a box and two tables, are placed

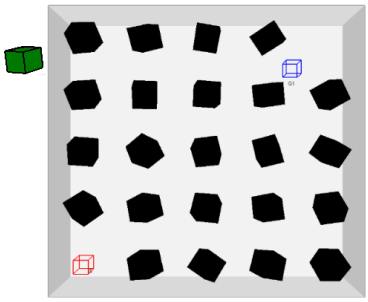
Algorithm 4 Planning n coarsely diverse paths

Input: G: A dense graph comprising of vertices set V and edges set E where $G = \{V, E\}$, Q: query to be solved from start to goal position, P: a set of n-distinct paths, p: vector array of path vertices.

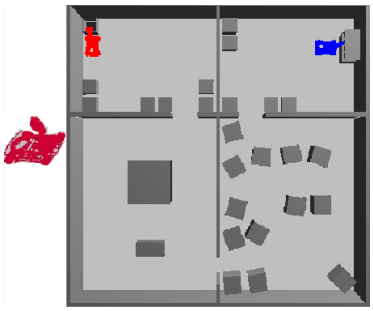
Let $n = 0$, $P \leftarrow \{\phi\}$, $p = 0$.
$S \leftarrow ConstructComplex(G)$; ◁ Refer Sec. 3.1
$TopologicalCollapse(S)$; ◁ Refer Sec. 3.1
$C \leftarrow IdentifyCriticalPoints(S)$; ◁ Refer Def. 4.1, Sec. 3.2
$F \leftarrow GetFeasiblePoints(S, C)$; ◁ Refer Def. 4.2
while $|C| > 0$ **do**
  if Q == false **then**
    $p = PlanPath(S)$
    if Q = true **then**
      $P = P \bigsqcup p$
      for all vertex $x \in$ p **do**
        $C = C/F^{-1}(x)$
        $y = F^{-1}(x)$
        $S = S/F(y)$
        Q $\leftarrow$ false
      $p.clear()$
    else
      No more path exists. Assign $C \leftarrow \phi$
$n = |P|$
for all $i = 0$ to $n - 1$ **do**
  for all $j = 1$ to n **do**
    if $\bigcup_{z \in P[i]} F^{-1}(z) \neq \bigcup_{z \in P[j]} F^{-1}(z)$ **then**
      list all $\mathcal{C}_{obst}$ covered by $P[i]$
    else
      $P = P/P[i]$
update $n = |P|$
return $\{P, n\}$

randomly in separate rooms, and the robot needs to pass through these obstacles to reach the goal position.

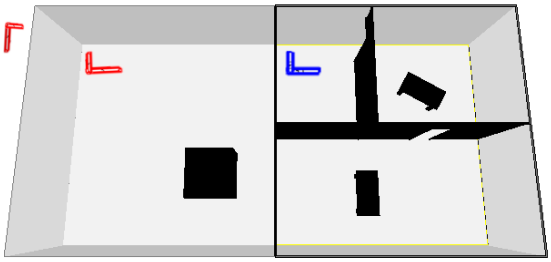
- **Kuka YouBot environment**: An 8 DOF robot in an environment with four different rooms, see Figure 4.2b. This robot is a simulation replica of Kuka YouBot [72]. The robot moves through different rooms within narrow passages and arrives at its destination, where it performs a task(grasps or puts an object down).



(a) **3D Cluttered environment**



(b) **Kuka YouBot environment**



(c) **House environment**

Figure 4.2: Environments Studied

4.1.3 Experimental Evaluations

This section provides a discussion on the results obtained for Uniform [68], Gaussian [28], and Bridge-Test [64] planners using our approach, and compared with the Voss method [135] which used the SPARS2 algorithm [48]. We average the result values over ten runs for topology map generation and ten runs for diverse path planning for each method in all environments.

4.1.3.1 Generating a Dense Topology Map

We generate a dense graph using the different sampling methods in all three environments. Table 4.1 shows the size of the maps needed to meet the sampling condition criteria from Section 3.1.

Table 4.1: Number of samples in the topology map

Environments	Uniform	Gaussian	Bridge-Test
3D Cluttered	5090	5918	6136
House	4985	6590	6504
Kuka YouBot	4908	6304	6601

Figure 4.3 show the edges formed on connecting nodes in a topology map for all planning methods.

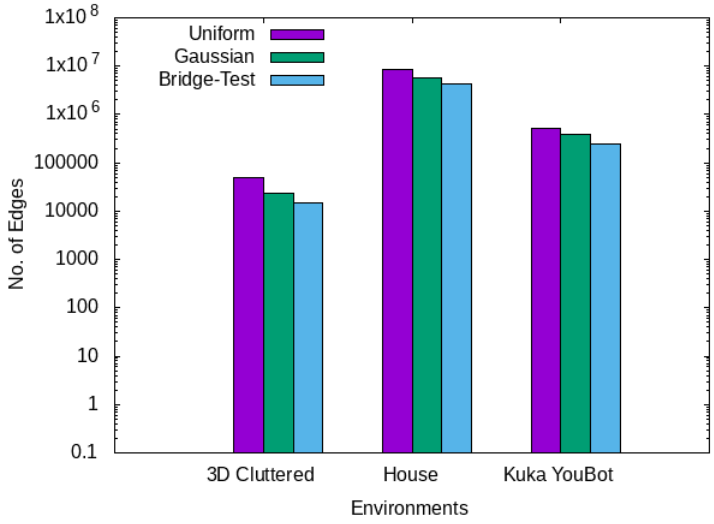


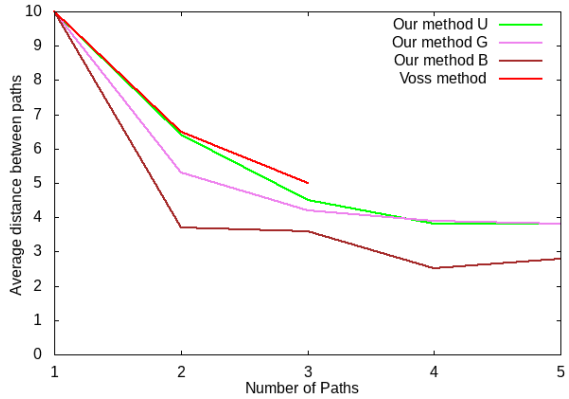
Figure 4.3: Total edges in the Topology maps

4.1.3.2 Number of Diverse Paths

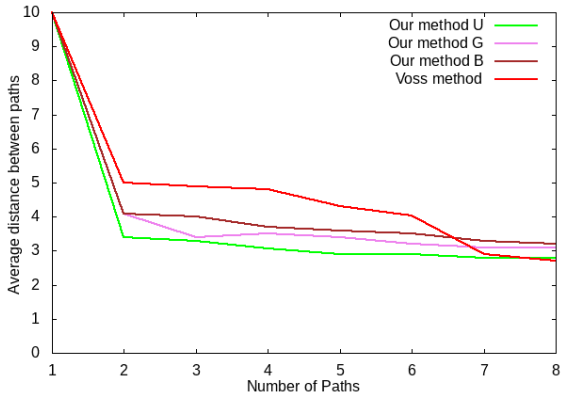
Figure 4.4 gives information about the number of distinct paths returned after running Algorithm 4 on our different environment scenarios. We used the Hausdorff distance metric to measure the distance between the pairs of routes. We observe that the Voss method returned fewer paths in the 3D cluttered environment (3) than our approach, which outputs five diverse pathways for all three planners. In the House environment, the Voss returns eight paths, while our method returns five in all planners' cases. We increase 100 units more for the path threshold T to obtain eight routes for all planners, as shown in Figure 4.4b. Interestingly, the average path length of our coarse paths in all planners' cases was lower than Voss, as visible in Figure 4.5b. Finally, in the Kuka YouBot environment, the Voss method failed to complete and returned 0 paths, while our approach returned four pathways for all three planners.

Voss method was unsuccessful due to its dependency on the SPARS2, which is limited to handling only planar and rigid body configurations (SE(2) and SE(3)), as discussed in [48]. We can also observe from Figure 4.4 that as the number of paths increases, the distance between routes decreases, and our method in all three planners maintained good diversity

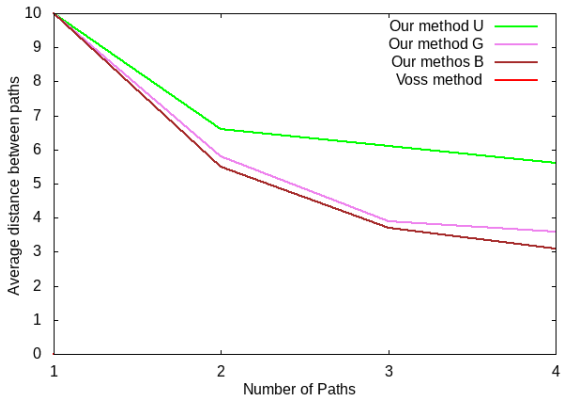
between obtained pathways.



(a) **3D Cluttered environment**



(b) **House environment**



(c) **Kuka Youbot environment**

Figure 4.4: We show a decrease in the average Hausdorff distance between paths as the number of coarsely diverse paths increases. For our method, U = Uniform topological planner, G = Gaussian topological planner, and B = Bridge-Test topological planner.

4.1.3.3 Setting a Threshold

We set a threshold for returned paths based on the range from the shortest diverse path length to a pathway of size T, where T is the mean of all shortest path lengths returned. It is due to the infinitely large number of diverse paths that are possible to generate in any environment depending on the mapping of routes to the plethora of critical points group combinations (detailed discussion available in Section 3.3.2). Table 4.2 to 4.4 gives information about our threshold and the diverse paths returned.

Table 4.2: Paths generated in 3D Cluttered environment

	Diverse Paths	Path length threshold
Uniform topological map	5	**300-400**
Gaussian topological map	5	350-500
Bridge-Test topological map	5	500-850
Voss method	3	N/A

Table 4.3: Paths generated in House environment

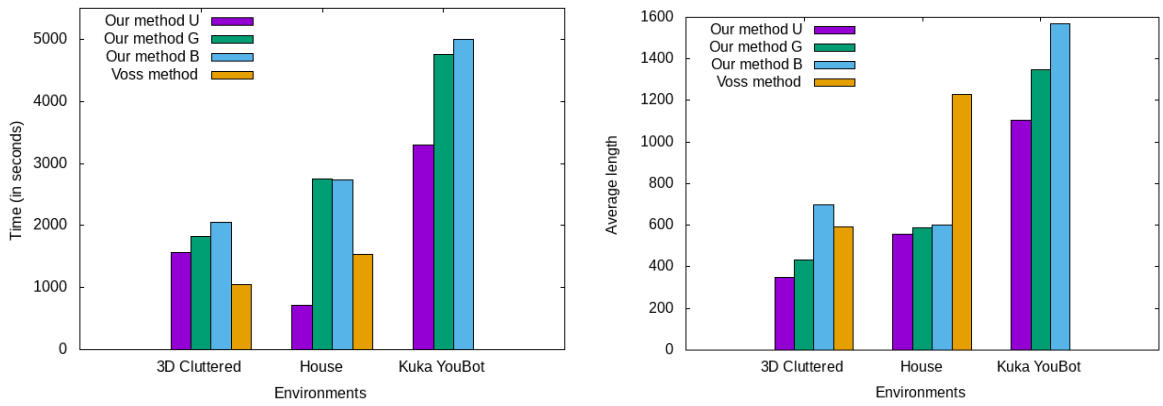
	Diverse Paths	Path length threshold
Uniform topological map	5	**522-600**
Gaussian topological map	5	500-700
Bridge-Test topological map	5	500-650
Voss method	8	N/A

Table 4.4: Paths generated in Kuka YouBot environment

	Diverse Paths	Path length threshold
Uniform topological map	4	**1020-1500**
Gaussian topological map	4	1050-2000
Bridge-Test topological map	4	1200-2500
Voss method	0	N/A

4.1.3.4 Computation Time and Average Path Length

In Figure 4.5a, we can see that the time needed to build the maps and generate the diverse paths differ in the environments. The Voss method uses less time in the 3D cluttered environment than our approach, but the Uniform topological planner outperforms it in the House environment. No result was recorded for Voss in the Kuka YouBot environment because it failed to complete. However, comparing the results of the topology methods in the Kuka YouBot environment, we see that the Uniform topology method uses less time than Bridge-Test and Gaussian and returns paths with a smaller average length, as seen in Figure 4.5b. We can postulate that the Uniform planner worked best with our topology approach due to the need to form a dense map in the environment as quickly as possible, a strength Uniform (Basic PRM) has over the other methods. Although Gaussian and Bridge-Test provide samples closer to the $\mathcal{C}_{obst}$, the approaches require enormous samples to cover the entire space, resulting in oversampling around the $\mathcal{C}_{obst}$. Thus, Gaussian and Bridge-Test operate effectively in maze-like situations when combined with our technique. A detailed description of the properties of these different planners is available in the literature review.

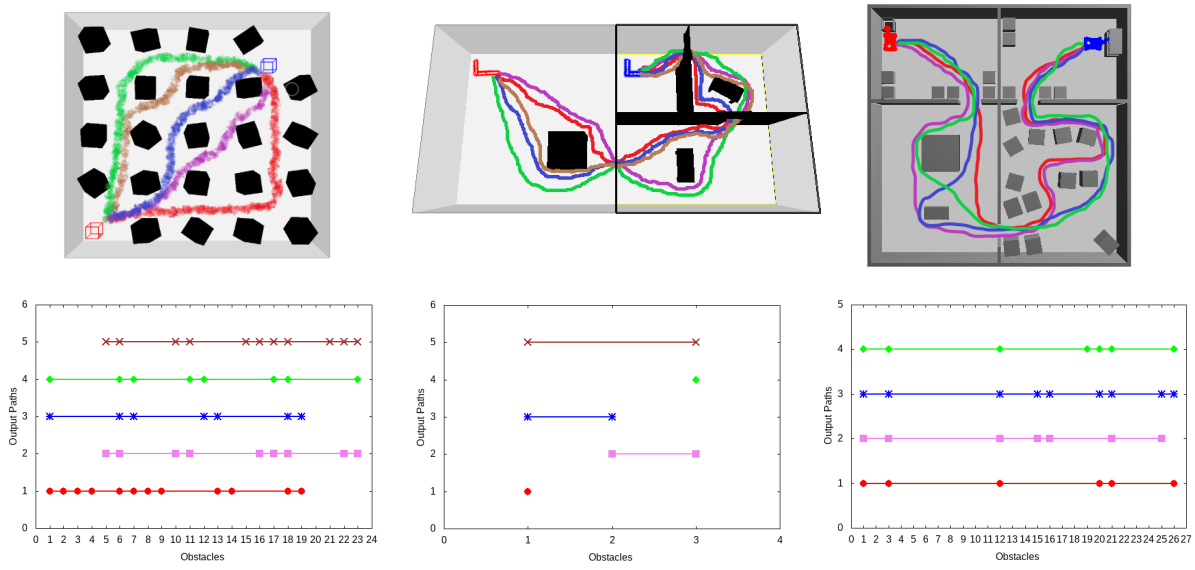


(a) **Computation Time** (b) **Average Path Length**

Figure 4.5: Computation Time and Average Path Length Comparisons.

4.1.3.5 Critical Points Analysis for Path Classification

We show some interesting analyses to aid an appreciation for the critical points and their role in identifying these diverse paths. Figures 4.6a to 4.6c show examples of pathways returned in the 3D Cluttered, House, and Kuka YouBot environments, showing the paths that map to a different set of critical points within a close distance of ϱ.



(a) **3D Cluttered** (b) **House** (c) **Kuka YouBot**

Figure 4.6: The plots below each respective environment show the set of critical points of $\mathcal{C}_{obst}$ that map to different coarsely diverse paths within a close distance ϱ for a Uniform topological planner. Different colors denote the output path's color in these environments, and the dots on each horizontal line represent the mapped critical points of the different $\mathcal{C}_{obst}$ for each path, from start to goal positions.

To better explain the relationship between the obstacles and the critical points, we create a plot that depicts the number of $\mathcal{C}_{obst}$ in the environment v/s the diverse paths that pass through the various groups of critical points of these obstacles, as seen in Figure 4.6. Recall that we define our diversity based on the distinct collections of critical points each path constitutes. The number of obstacles in the 3D Cluttered environment is 23, with 170 identified critical points. The House environment has 3 $\mathcal{C}_{obst}$, with 95 identified critical points, and the Kuka YouBot environment has 26 obstacles, with 210 identified critical points. We also previously proved in *Proposition 1* that our critical points determine the number of coarsely diverse paths we can generate. In summary, we have successfully provided a new methodology to extract coarsely diverse paths from the topological and geometric information of the $\mathcal{C}_{space}$ and compared it with a baseline method. Our approach has successfully shown that distinct routes of shorter lengths can be possible in high-dimensional spaces.

Hence, we can conclude that this work provides a new way to categorize paths into homotopy classes using the geometric properties of $\mathcal{C}_{obst}$. It also provides the set of diverse paths computed during the single roadmap generation, thus, avoiding re-planning overhead.

4.2 Fault-tolerant Motion Planning

Designing an autonomous, fault-tolerant manipulator [84, 1] requires identifying configurations that can withstand failures with minimal displacement of constraints while maintaining the manipulator's dependability in the event of partial mechanism failures. Such manipulators are especially useful in complex, confined, and dynamic environments such as disaster areas, nuclear disposal sites, and deep-sea or space exploration. Since these environments are often uncharted and dynamic, positioning robotic manipulators in optimal configurations is critical for high fault tolerance and performance. Choosing a solution path with fewer configurations can make fault tolerance more practical since it requires minor adjustments in the relative configuration node when there are changes in the planning space.

When a motion planning attempt fails, the goal is to identify the cause of the failure in order to take the appropriate recovery action [75, 29]. Geometric and topological methods have been used to provide proof of disconnection or infeasibility by decomposing the $\mathcal{C}_{free}$ space, using alpha shapes and other approximations of obstacle space [87, 80, 130]. However, the focus of this work is not on finding the cause of failures but on reacting or recovering

from the changes in the system that occurred due to failures.

This work introduces a novel algorithm for near-optimal path planning of a robotic system that focuses on fault tolerance and recovery applications. To reduce the number of re-configurations in finding a feasible path, we introduce the novel Minimal Path Violation (MPV) methodology. MPV ensures that there exists a route that is collision-free and connects to start and goal positions with minimum re-configurations required. Firstly, the algorithm constructs the complete graph of the environment and calculates the coarsely diverse routes discussed in Section 4.1. Secondly, it prioritizes these paths using three ranking measurements, i.e., path cost, the node's visibility, and the edge's expansiveness. Thirdly in the event of any change in the planning space, the proposed algorithm suggests an alternate path without recomputing topological information, thereby saving computational costs. The ranking measures allow the identification of the adjacent best route in case of a path failure, which does not get easily invalidated due to changes in the environment. Figure 4.7 shows the workflow of the proposed algorithm.

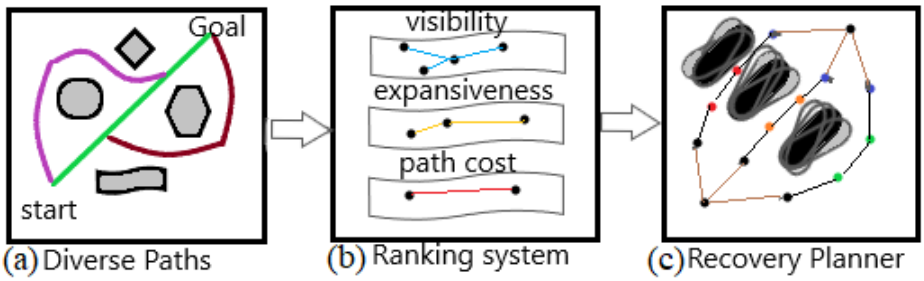


Figure 4.7: The algorithm generates diverse paths and ranks them using node visibility, edge expansiveness, and path cost. Using this information prevents the invalidation of a path with a minimal number of configurations. The points in the (c) are in red, orange, and green colors around obstacles based on the configuration violation detected in the new configuration space.

4.2.1 Formal Definitions

We present the Minimal Path Violation framework and three ranking conditions utilized to prioritize the diverse paths for fault-tolerance operation.

4.2.1.1 Minimal Path Violation (MPV)

Given a sampled graph G of vertices V and edges E, i.e., $G = \{V, E\}$, such that set V represents the configurations in the closure of $\mathcal{C}_{free}$ and set E represents the collision-free connections between two endpoints $u, v \in V$. Let $\mathbb{P}$ be a set of diverse paths, and $\mathbb{P}_V$ and $\mathbb{P}_E$ define the vertices and edges covered by these paths. We say path p is feasible if $\mathbb{P}_V \subseteq V$ and $\mathbb{P}_E \subseteq E$. Let Q' be the set of all violated configurations in changed $\mathcal{C}_{space}$, such that $Q' \subseteq \mathbb{P}_\mathbb{V}$. We define the Minimal Path Violation between two endpoints $s, g \in \mathcal{C}_{free}$ as below.

Definition 4.4 *(**Minimal Path Violation**): A path p' is collision-free and minimal (in the changed $\mathcal{C}_{space}$) if and only if $Q' \cap \mathbb{P}_\mathbb{V}(p') = \emptyset$ and $|\mathbb{P}_\mathbb{V}(p')| - |\mathbb{P}_\mathbb{V}(p)|$ is minimum $\forall p' \in \mathbb{P}$, where $\mathbb{P}_\mathbb{V}(p)$ is the set of vertices covered by p, p is the straight line path between s and g, and $|.|$ represents the cardinality of the set.*

Figure 4.8 shows a general idea of MPV. We solve the MPV problem in the changed $\mathcal{C}_{space}$ using our ranking measure that assures the minimum number of vertices gets connected to find a pathway between the start and goal position. Our ranking system takes the node's visibility, edge's expansiveness, and path cost to respond quickly to a changing environment with the next shortest pathway.

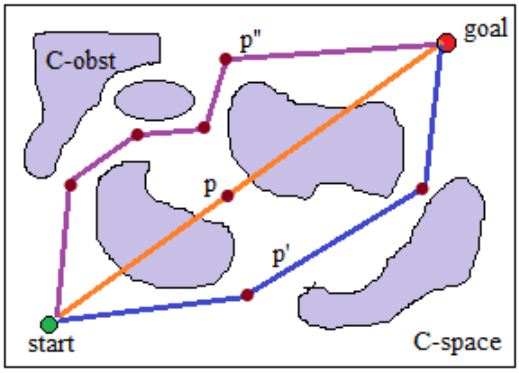


Figure 4.8: An illustration of an MPV problem in a 2D $\mathcal{C}_{space}$ where p is an unfeasible straight line path between s and g. At least one configuration needs to be displaced to avoid the invalidation of the route p. The solution to the MPV problem requires finding the next best route that connects the start and goal with the minimal number of nodes' re-configuration. Our ranking system solves this problem by prioritizing p' over p'' such that the number of nodes difference between p and p' is minimum while it is also the shortest path.

Consider S as the $\mathcal{C}_{free}$ topology-approximated roadmap with n diverse paths in it, where $S \subseteq G$ and $|\mathbb{P}| = n$. A roadmap S is feasible if the number of valid paths is higher

than the number of invalid paths in the new $\mathcal{C}_{space}$, i.e., $(n-t) > t$, where t is the count of the number of invalid paths. We use the bound of MPV, as stated in Theorem 4.1, to assess the validity of roadmap S.

Theorem 4.1 *Given a roadmap S, with m valid configurations and r is the lower bound of intermediate configuration nodes required to connect a path between source S_a and destination S_b in the $\mathcal{C}_{space}$, where $S_a, S_b \in \mathcal{C}_{free}$; $m > r$. Then, the bound for finding the Minimal Path Violation in S is $\frac{r*(m-r)!*r!}{m!}$.*

Proof: Let L be the set of configurations of the shortest path p, i.e., $|L| = r$. To invalidate path p in $\mathcal{C}_{space}$, the vertices of L should be arranged so that at least one configuration $c \in L$ is either violated or collides with $\mathcal{C}_{obst}$. So, the number of permutations of getting a configuration violation that invalidates the path is $P(r,1) = r$, refer [139]. For m configurations in S, the combination of finding the diverse paths with r configurations is $C(m,r) = \frac{m!}{(m-r)!r!}$, refer [137]. Hence, the bound for finding the Minimal Path Violation for S can be given by $\frac{P(r,1)}{C(m,r)} = \frac{r*(m-r)!*r!}{m!}$. □

We evaluate the feasibility of the ranked set at the time of recovery planning (in Alg. 6) using Theorem4.1.

4.2.1.2 Ranking Measures

Here, we discuss our ranking measures to prioritize the diverse paths based on the topology properties of the $\mathcal{C}_{space}$, i.e., path cost, node visibility, and edge expansiveness. We perform these measures on a topology-approximate roadmap defined in our prior work [126, 124]. From this topology map S, we compute the diverse paths set $\mathbb{P}$ and perform a ranking of the routes via the conditions described in Proposition 1 - 3.

Proposition 4.3 ***(Path cost)**: Taking $\mathbb{P}$ as the set of diverse paths, then a path p is given priority if it has the shortest length compared to the $n-t$ paths.*

Recall that n is the total number of diverse paths in $\mathbb{P}$ and t is the total number of invalid pathways. Here, the general idea of sorting the routes based on their lengths in $\mathbb{P}$ is considered, thus, giving priority to the shortest path with feasibility, i.e., from the shortest

path length to the longest. A route becomes the next best alternative if it connects to fewer vertices and edges than the valid paths in the modified $\mathcal{C}_{space}$. □

Proposition 4.4 ***(Node visibility)****: Let K define the limit of maximum visibility. If a vertex $v \in V$ has j connected neighbors where $j \leq K$ and $j \neq 0$. Then, we define the priority of v with the rank score achieved on $K - j$, i.e., the higher the rank score means the highest priority.*

Here, K defines the maximum number of unique neighbors a vertex (or a configuration node) can connect, and j is the number of successful connections made for K connection attempts. The visibility of v depends on the number of unsuccessful connection attempts made and computed by the parameter $l = K - j$ where $l > 0$. The larger the value of l, the lower the visibility of the v in the $\mathcal{C}_{space}$ (existence in narrow regions). We prioritize the low visibility nodes because they are closer to $\mathcal{C}_{obst}$, and the path connecting it will pass at proximity to $\mathcal{C}_{obst}$. Moreover, the chances of finding an alternate route through low-visibility nodes are less than through high-visibility nodes. So, the process of discarding low-visibility nodes when they become invalid and transitioning to higher-visibility nodes during re-configuration becomes simpler through ranking. As shown in Figure 4.9, x has a low visibility than y and z. It is, thus, given higher priority than y and then z. □

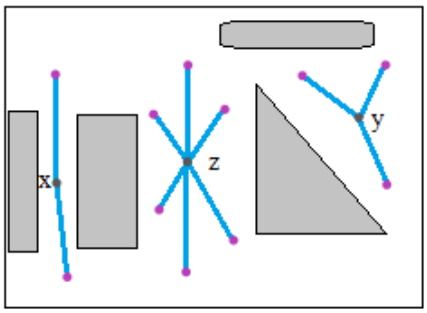


Figure 4.9: Visibility ranks of vertices. The vertices with low visibility or lying closer to $\mathcal{C}_{obst}$, such as x, are ranked higher than vertices with high visibility, such as y and z, respectively.

Proposition 4.5 ***(Edge expansiveness)****: Let S_a and S_b be vertices in a roadmap S of the $\mathcal{C}_{free}$ topology-approximation graph G. Suppose the shortest unfeasible path between S_a and S_b in the $\mathcal{C}_{space}$ has length D. If an edge β connects two vertices between S_a and S_b and has length $h \leq D$, then the priority of β is determined by the rank score $D - h$. Specifically, the smaller the rank score, the higher the priority of the edge β.*

We can justify the proposition as follows. First, consider a straight line in $\mathcal{C}_{space}$ connecting S_a and S_b, which need not be collision-free. The collision-free edges between the vertices in the roadmap S determine the safe route for the robot in $\mathcal{C}_{free}$. Suppose the length of edges connecting vertices between S_a and S_b is at most h_{max}. Then, the collision-free edges of length h closer to D will need fewer nodes to form a path than those further away. Hence, longer collision-free edges are more desirable as they reduce the number of nodes in the pathway. This property can help search for the next feasible edge with minimum displacement from the same vertex. In particular, the priority of an edge β connecting vertices between S_a and S_b is based on its rank score $D - h$. Since D is the length of the shortest unfeasible path between S_a and S_b, edges with smaller lengths $h < h_{max}$ have higher rank scores and lower priority.

To illustrate, consider Figure 4.10, where the length of the shortest unfeasible path between S_a and S_b is D, and h_U is the edge between S_a and S_b, h_V is between q to S_b, and h_W is between r to S_b along paths U, V, and W, respectively. Then, during ranking, h_U has higher priority than h_V and h_W, and the rank score is $D-h_U$. Also, in general, D always takes length as h_U between a start and goal position. Therefore, the edge expansiveness property helps efficiently find a feasible path between two vertices in the roadmap by prioritizing longer, collision-free edges. □

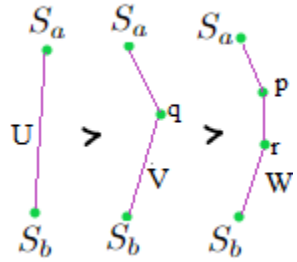


Figure 4.10: Ranking of edge expansiveness. It prioritizes longer collision-free edges in the paths, such as U, over short edges in V and W.

When ranking paths, the priority sequence is determined by path cost followed by node visibility and edge expansiveness.

4.2.2 Algorithmic Implementation

We perform two associated steps to overcome/recover from the faulty scenario. First, we apply propositions 4.3 - 4.5 to rank our collision-free coarsely-diverse paths. Second, we use the ranking information to find a valid pathway in the changed $\mathcal{C}_{space}$ realizing MPV framework. These two algorithms are performed as the post-processing steps to the already approximated $\mathcal{C}_{free}$-topology graph [126, 124].

4.2.2.1 Ranking diverse paths

Algorithm 5 aims to rank and return a set of coarsely diverse paths for a given dense graph G while solving a query from start to goal position. The algorithm uses a combination of topological and geometric information abstracted from G, as provided by methods introduced in previous sections 3.1 and 3.2.

To compute the diverse paths, the algorithm employs the *GetPath* method (lines 2-10), which generates a set $\mathbb{P}$ of n-distinct paths in S, where the parameter n controls the number of pathways to identify. Next, the algorithm computes the visibility of the configuration nodes in S (from Proposition 4.4), using the *CountNeighbors* method (lines 11-17). The visibility information prioritizes the nodes that are more likely to be part of high-quality paths.

To further prioritize the high-quality paths, the algorithm creates a priority queue of collision-free edges with longer lengths (as per Proposition 4.5) in lines 18-21. The algorithm then sorts the set of diverse paths in $\mathbb{P}$ based on their path lengths (as per Proposition 4.3), using the *Sort* method (line 24). The paths with the lowest cost are ranked first, with ties broken by giving importance to the number of high-priority nodes and then high-priority edges (lines 25-29). The algorithm outputs a set R of ranked paths for solving query from the same start to a goal position in changed $\mathcal{C}_{space}$. Overall, Algorithm 5 combines our various methods and techniques [126, 124, 122] to efficiently rank and generate a diverse set of high-quality paths from a given dense graph.

Algorithm 5 Ranking n coarsely diverse paths

Input: G: A dense graph comprising of vertices set V and edges set E where $G = \{V, E\}$, Q: query to be solved from start to goal position, $\mathbb{P}$: set of n-distinct paths, p: vector array of path vertices, n: number of paths to be computed, K: node-reachability parameter, D: distance b/w start and goal position, R: set of ranked paths.

Let $i = 0$, $\mathbb{P} \leftarrow NULL$, $p = 0$.
$T = GetTopologyGraph(G)$; ◁ Refer Sec. 3.1
$M = GetCriticalPoints(T)$; ◁ Refer Sec. 3.2
$S = T \bigcup M$
while $i < n$ **do**
    Q = false
    $p = GetPath(S)$; ◁ Refer Sec. 4.1
    if Q = true **then**
        $\mathbb{P} = \mathbb{P} \bigsqcup p$
        i = $|\mathbb{P}|$
$q \leftarrow NULL$
for all nodes $x \in S(V)$ **do**
    $j = CountNeighbors(x, S(V))$; ◁ Proposition 4.4
    if $j \neq K$ **and** $K - j > 0$ **then**
        $PushFront(q, x)$;
    else if $j == K$ **then**
        $PushBack(q, x)$;
$e \leftarrow NULL$
for all edges $y \in S(E)$ **do**
    $h = D - length(y)$; ◁ Proposition 4.5
    $e.push(h, y)$;
$r \leftarrow NULL$
Sort($\mathbb{P}$); ◁ Proposition 4.3
$p_0 \leftarrow NULL$
for all $p \in \mathbb{P}$ **do**
    if $p_0 \neq NULL$ and $cost(p_0) == cost(p)$ **then**
        $r = Rank(\{p_0, p\}, q, e)$;
    else
        $r.insert(p)$
    $R = R \bigcup r$
    $p_0 = p$
return R

4.2.2.2 Minimal Path Violation planning

Algorithm 6 considers the ranked paths set $R \subset S$ and the environment map X of the new $\mathcal{C}_{space}$ as the input, where $dim(S) \neq dim(X)$. Here, dim calculates the dimension of the space. The $Projection$ method maps the configuration nodes of $dim(S)$ to $dim(X)$ using

transformation matrix I in lines 3-6.

$$I_{i*j} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}; \quad i = dim(S), j = dim(X). \tag{4.1}$$

The algorithm evaluates the validity of the paths in the set R using the map X in line 7. The pathway is pruned from R if the current shortest path is invalid due to new changes in the $\mathcal{C}_{space}$, and the alternate shortest route from R gets selected for validation in lines 8-10. When the number of valid nodes decreases in $R \subset S$, the number of invalid paths increases, making the roadmap S unfeasible, i.e., $R \rightarrow null$. The increase in the probability of MPV (from Theorem 4.1) for R decreases the chances of finding a feasible solution in X which tends towards 0 (line 10). The process continues until a near-optimal pathway is found in X, else the algorithm calls a complete planner (like PRM [68], RRT [78], e.t.c.) to find a path due to the discretization of the space.

Algorithm 6 Recovery path planner

Input: X: a new $\mathcal{C}_{space}$, S_a: start position, S_b: goal position, p: vector array of path vertices, R: set of ranked paths, Γ: a complete planner.

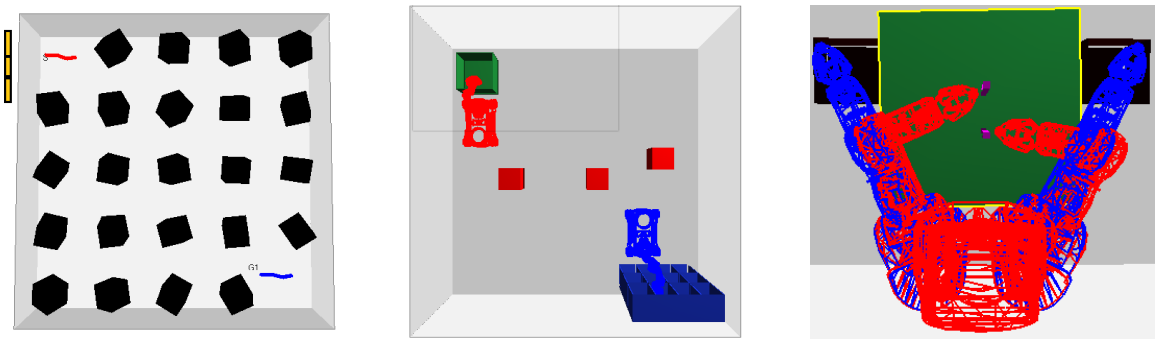
1: $x = dim(X)$;
2: $R = GetRankedPaths()$; ◁ Algorithm 5
3: $S_a \leftarrow Projection(S_a, x); S_b \leftarrow Projection(S_b, x)$
4: **while** S_a and S_b not connected **do**
5: **for all** $p \in R$ **do**
6: $p \leftarrow Projection(p, x)$;
7: **if** p is not valid in X **then**
8: $remove(R, p)$;
9: **if** R not null **then**
10: $p = SelectNextPath(R, X)$; ◁ Theorem 4.1
11: **else**
12: Calls $\Gamma(S_a, S_b)$
13: **else**
14: Connect p with S_a and S_b
15: **return** p

Using Algorithm 5 to rank our diverse path and Algorithm 6 with MPV for recovery, we reduce the re-planning time and computation cost that we evaluate in Section 4.2.4.

4.2.3 Experimental setup

We executed experiments on a Dell Alienware Aurora desktop machine running the Ubuntu 20.4 LTS operating system. This setup is different from our previously evaluated research work. We performed experiments in 3 separate environments, as shown in Figure 4.11. Figure 4.12 shows the start and goal positions in red and blue color, respectively.

- **3D Cluttered environment**: Obstacles are cluttered around the room as shown in Figure 4.11a. The robot has to traverse through these obstacles successfully to reach its goal. The robots are 3 DOFs and 2 DOFs articulated linkages.
- **Pick and Place environment**: A shelf and an open box are placed diagonally to each other, and three small boxes are scattered randomly in the space, see Figure 4.11b. The robots are 7 DOFs and 10 DOFs Kuka YouBots [72].
- **Object displacement environment**: The surrounding has a table with two boxes placed on either side of the table. The robot should move the stick from the table and drop it into the box using its hands, as shown in Figure 4.11c. The robots are 14 DOFs and 28 DOFs PR2 [140] with a fixed base.



(a) **3D Cluttered** (b) **Pick and Place** (c) **Object displacement**

Figure 4.11: Environments Studied

4.2.4 Experimental Outcomes

In this section, we discuss the performance of our method in the modified $\mathcal{C}_{space}$ and compare its results with optimal planners, like LazyPRM^* [61], RRT^*-Connect [69], and Informed RRT^*-Connect [86]. Here, we take $K = 5$, whereas D is calculated empirically

within the algorithm for ranking conditions. We performed a total of 225 executions to generate roadmaps for each planner in all three environments, and the results were averaged over 15 runs to get standard deviation values.

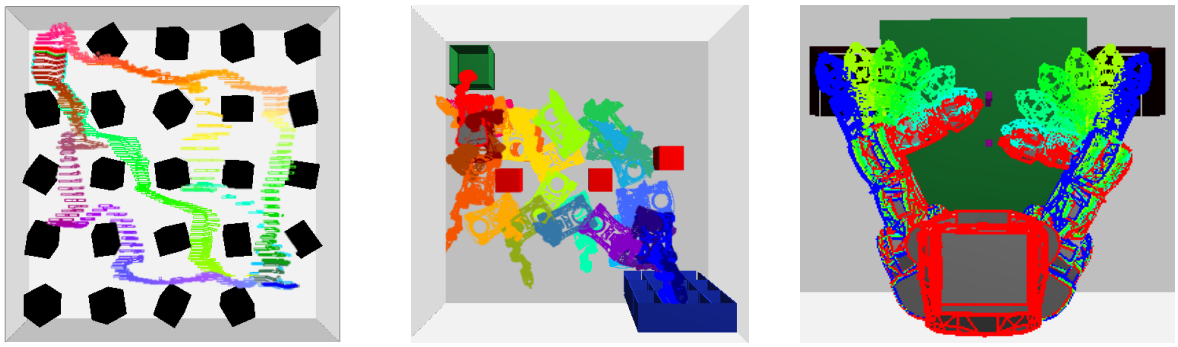
4.2.4.1 Solving MPV problem in changed $\mathcal{C}_{space}$

We define a "change" in the $\mathcal{C}_{space}$ as the inclusion or exclusion of $\mathcal{C}_{obst}$ that invalidates the existing solution. So, when these changes occur in the $\mathcal{C}_{space}$, we apply our MPV framework to find an alternate path during recovery planning.

Prioritising coarsely-diverse paths: We generate topology maps and diverse paths for all three environments using 3 DOF articulated linkage robot, 10 DOF Kuka YouBot, and 28 DOF PR2 robot (two arms), respectively. Figure 4.12 shows an example of coarsely-diverse pathways in each environment, and Table 4.5 shows the size of the rank set, the range of path cost, nodes, and edges for the coarsely diverse paths derived from Algorithm 5. This table gives the information needed for MPV, including the minimum and the maximum number of nodes/edges present in the paths in these environments.

Table 4.5: Ranked coarsely diverse paths

Environments	**Total paths**	**Cost**	N_{nodes}	N_{edges}
Cluttered	84	280-546	10-36	14-22
Pick and Place	3	1584-1901	30-96	38-52
Object displacement	2	504-630	12-18	11-17



(a) **3D Cluttered** (b) **Pick and Place** (c) **Object displacement**

Figure 4.12: An illustration of coarsely-diverse paths planned in each environment. For the 3D Cluttered environment in (a), we have shown only four routes for better visualization.

4.2.4.2 Fault-tolerant planning in changed $\mathcal{C}_{space}$

Initially, we generate and rank various paths for robots with 3, 10, and 28 degrees of freedom (DOF) in their respective environments. We then introduce dynamic behavior in these environments by modifying the obstacles' positions, constraining the manipulator's joint angles, or limiting the query evaluation time, thus, causing the existing shortest paths to become unfeasible and the planner to fail to identify the same route. Consequently, our approach identifies an alternative pathway with fewer DOF to serve as a recovery measure in these environments. The aim is to find a fault-tolerant path that becomes feasible by reducing the DOF of the robot in consideration (a subset of the original problem) for the detected fault in the initial environment.

Cluttered environment: We add two new obstacles and plan a path using a 2 DOF articulated linkage robot to recover from faulty $\mathcal{C}_{space}$. We see that the placement of the two recent objects invalidates the current shortest path from Figure 4.13a, and as a result, our planner looks for the next possible pathway to connect start and goal positions as shown in Figure 4.13d.

Pick and Place environment: We add one new obstacle and restrict the joint angle of one of the links (introduces extra computation time). The initial shortest path of 10 DOF Kuka YouBot is shown in Figure 4.13b and the fault-tolerant pathway planned using 7 DOF Kuka YouBot is shown in Figure 4.13e.

Object displacement environment: We restrict the forearm movement of the left hand and limit the query evaluation to 10 seconds. The faulty scenario of the PR2 robot is captured in Figure 4.13c, where it is unable to find a path due to restricted movement of the left hand, and Figure 4.13f shows the recovery path planned using 14 DOF PR2 robot (one arm). In the changed $\mathcal{C}_{space}$, the right-hand picks both the sticks, which were initially picked separately by each hand. Hence, we have two paths planned for the right hand that performs one stick displacement initially and two sticks displacement for the changed $\mathcal{C}_{space}$.

Table 4.6 shows the number of configuration nodes in the shortest path ($Path_A$) and the new shortest path ($Path_B$) in changed $\mathcal{C}_{space}$ at the time of recovery planning. $Path_A$

refers to the initial route before invalidation, and $Path_B$ is the new path obtained using the ranking measures that solve the MPV problem.

Table 4.6: MPV solution in new $\mathcal{C}_{space}$

Environments	$Path_A$		$Path_B$	
	Nodes	Cost	Nodes	Cost
Cluttered	$10^{\pm 10.76}$	$280^{\pm 2.79}$	$13^{\pm 9.45}$	$286^{\pm 3.4}$
Pick and Place	$30^{\pm 5.11}$	$1584^{\pm 11.23}$	$36^{\pm 2.76}$	$1848^{\pm 9.83}$
Object displacement	$12^{\pm 1.34}$	$504^{\pm 1.0}$	$18^{\pm 1.98}$	$630^{\pm 0.67}$

From Tables 4.5 and 4.6, we observe that the re-configuration of nodes during recovery planning maintains the minimality of the MPV problem. Our method selects the adjacent best route within the ranked paths with a minimum re-configuration of 2 to 6 nodes in all three environments. Hence, we can conclude that our method efficiently solved the MPV problem to find a valid path by using the ranking measure that guaranteed minimum node re-configuration from the invalid pathways.

Table 4.7 shows the displacement distance between the initial and re-planned paths in the changed $\mathcal{C}_{space}$. We compare the results of our method with optimal planners LazyPRM^*, RRT^*-Connect, and Informed RRT^*-Connect. We use the Hausdorff distance to measure the distance between two paths. We observed that our method identifies a feasible pathway in new $\mathcal{C}_{space}$ with the smallest displacement distance between initial and re-planned routes compared to the paths generated by other methods in all three environments.

Table 4.7: Distance b/w initial and re-planned paths

Environments	**Our Approach**	**Lazy**PRM^*	RRT^***-Connect**	**Informed** RRT^***-Connect**
Cluttered	$1.07^{\pm 0.08}$	$2.68^{\pm 1.06}$	$2.02^{\pm 0.95}$	$1.27^{\pm 0.27}$
Pick and Place	$13.39^{\pm 0.14}$	$18.85^{\pm 1.99}$	$16.63^{\pm 1.32}$	DNF
Object displacement	$0.01^{\pm 0.007}$	DNF	$0.08^{\pm 2.65}$	DNF

We conclude that using the MPV probability and ranked diverse paths information can aid in finding the next feasible solution with minimum re-configurations.

4.2.4.3 Analysing the Recovery Solution

The goal is to ensure a nearly optimal solution while recovering from a faulty scenario. We evaluate our recovery solution against optimal planners in time, cost, and nodes

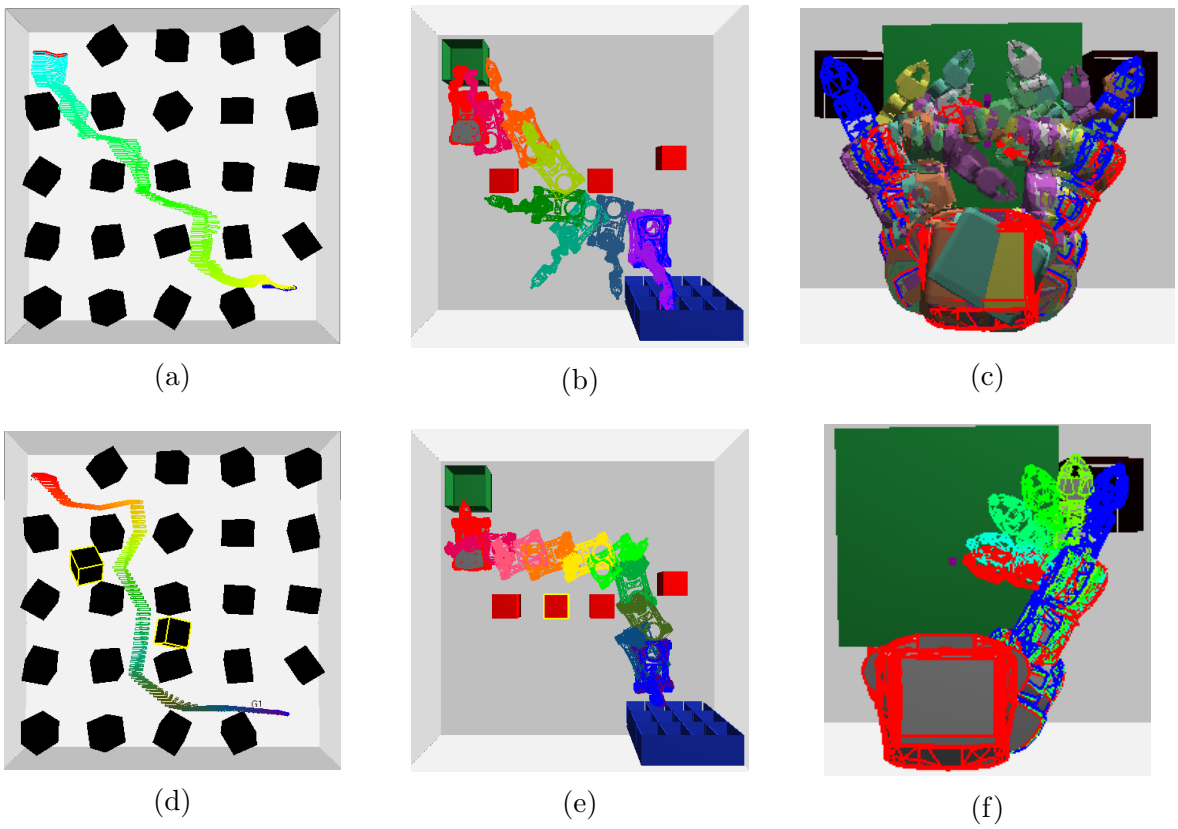
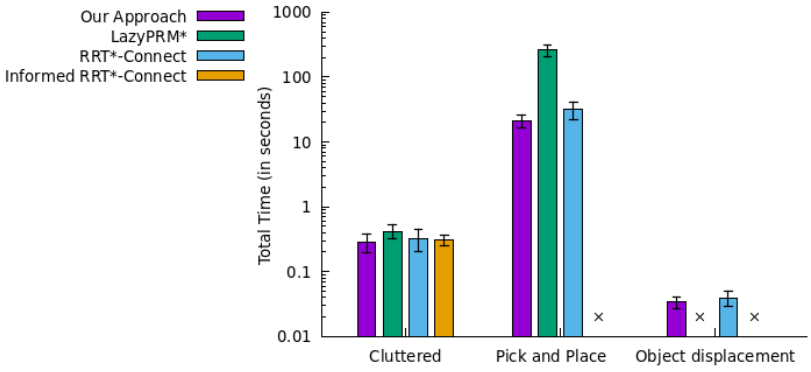
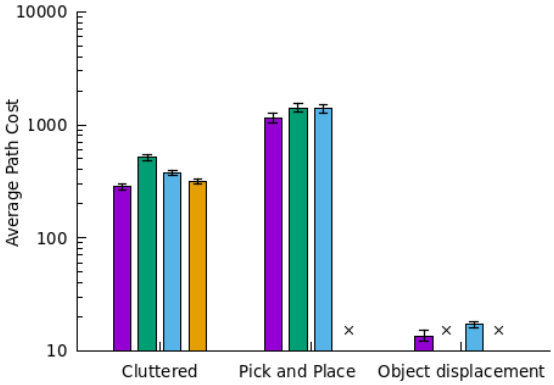


Figure 4.13: Figures (a) and (b) show the initial paths for 3 DOF articulated linkage robot and 10 DOF Kuka YouBot, respectively. Figure (c) shows the faulty scenario for the PR2 robot with a restricted left hand. Figures (d), (e), and (f) show fault-tolerant paths for 2 DOF articulated linkage robot, 7 DOF Kuka YouBot, and 14 DOF PR2 robot, respectively.

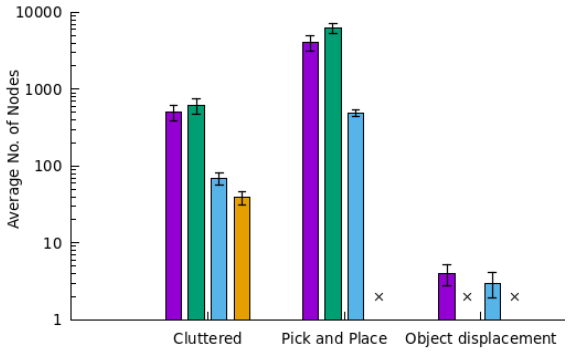
to analyze resource consumption. Using Algorithm 5 and 6, we compute diverse paths and re-use information across three environments through a ranking system. Our planner's computation time includes roadmap generation, ranking, and recovery evaluation. We compare our results with LazyPRM^*, RRT^*-Connect, and Informed RRT^*-Connect, in initial and changed $\mathcal{C}_{space}$, taking average values for planned and re-planned roadmap nodes and path cost. Our method, as seen in Figure 4.14, outperforms in computation time and path cost without the need to re-generate a roadmap while generating more nodes than RRT^*-Connect and Informed RRT^*-Connect due to building a complete graph of $\mathcal{C}_{space}$. LazyPRM^* failed to find a path for the PR2 robot due to query resolution, whereas, Informed RRT^*-Connect fails to provide valid configurations within the ellipsoidal curve for Kuka YouBot and PR2 robots (computed from [58]).



(a) **Computation Time**



(b) **Path Cost**

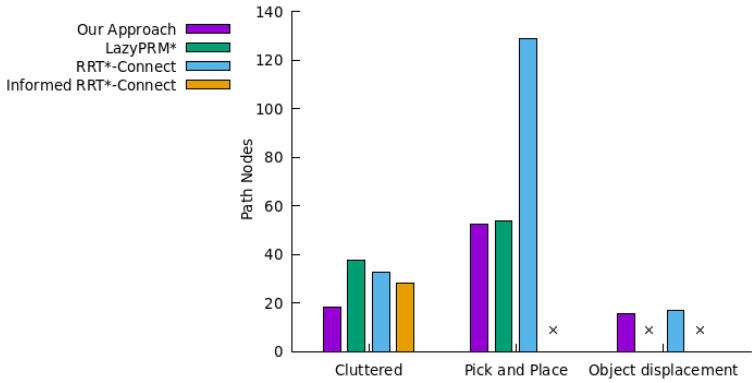


(c) **Total Nodes**

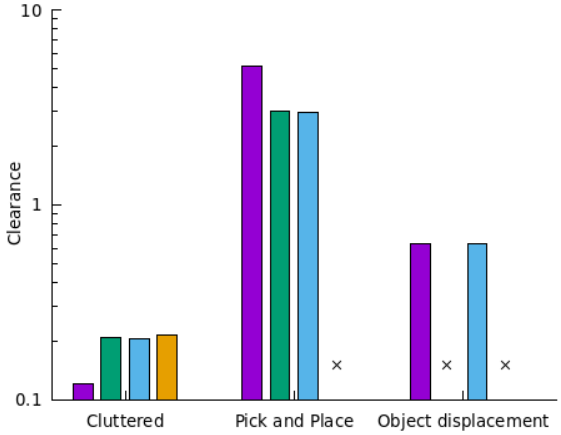
Figure 4.14: The computation time comprises the total time taken to compute a path in the initial and the new $\mathcal{C}_{space}$. We take the average of planned and re-planned roadmap values for the path cost and the total number of nodes. The "x" in the plots refers to DNF (did not finish).

Path Metric Evaluation: We also evaluate the path metrics of the initial and new paths by comparing the average number of nodes in the routes, the average minimum clearance from the $\mathcal{C}_{obst}$, and the deviation of the new pathway from the initial path using smoothness measure (in radians). Figures 4.15a and 4.15c show that our method-planned optimal pathways have fewer nodes and minor deviations in all environments compared to other planners. In Figure 4.15b, we noticed a smaller clearance in the Cluttered environment and a higher clearance value for Kuka YouBot and PR2 robot for our planner, which shows its ability to adapt to safety measures in high-dimensional $\mathcal{C}_{space}$.

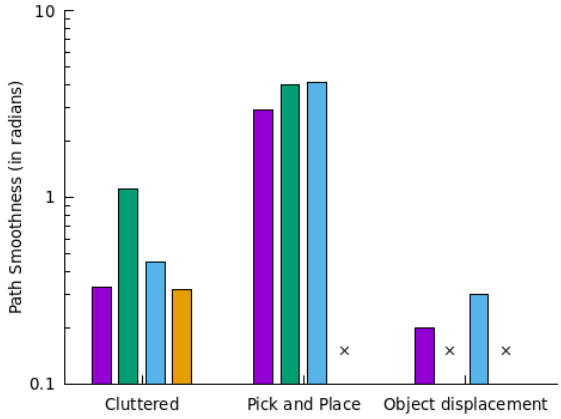
As a result, we conclude that our method provides a novel way for recovery path planning by utilizing the ranking measure of coarsely-diverse paths and the Minimal Path Violation framework to overcome the re-planning time and computation cost overhead for



(a) **Path Nodes**



(b) **Minimum Clearance**



(c) **Path Smoothness**

Figure 4.15: The plots show an average of the planned and re-planned roadmap values for the path nodes, clearance from obstacles, and path smoothness (in radians). The "x" in the plots refers to DNF.

fault-tolerant scenarios.

In conclusion, we have presented two innovative approaches that leverage the information of the topology roadmap to address the challenges of path or motion failures during path planning. These methods demonstrate an intelligent way to utilize the configuration space's topology and exhibit consistent performance across various robot types and environments.

CHAPTER 5

Application to Computational Biology

Molecular modeling is a field that deals with predicting the binding conformation of protein-ligand or protein-protein complexes. This problem commonly gets solved through computational methods that accurately predict the 3D structure of the bio-molecule upon binding to the target protein receptor. It is a crucial aspect of drug discovery and development and requires precision and accuracy in its approach. This chapter discusses the application of our motion planning technique to study and analyze the binding behavior of bio-molecules during their interaction with others which leads to their structural changes [125, 119, 120].

In this chapter, "conformation space" is used interchangeably with "configuration space." The biomolecule's conformation has similar representations as the robot's configuration. We treat macro-molecules as stationary objects and view a binding bio-molecule, such as a ligand or Intrinsically Disordered Protein (IDP), as a moving articulated linkage robot.

5.1 Protein-ligand Interactions

Protein-ligand interactions are crucial to a wide range of biological activities and functions in any organism, including cell metabolism, signal transduction, muscle contraction, and immune systems. Many essential cellular processes, e.g., controlling the functioning of enzymes, transport, and most regulatory mechanisms, rely on physical interactions between proteins. Therefore, protein-ligand interactions network analysis is essential to gain comprehensive knowledge of the control mechanism and its organization in a living cell.

Analyzing and extracting useful information from the molecular surface of bio-molecules is a fundamental problem in structural biology [150]. These surfaces contain essential biological information that helps us understand properties such as the geometrical organization of interacting residues, precise identification of the borders of each interaction site, energy potential at interaction sites that allow for strong versus weak binding, and the locations where artificial molecules (e.g., drugs) can best bind.

Understanding these properties has life-saving biological implications, including aiding in therapeutic drugs, vaccines, and point-of-care technology development. However, predicting these protein-ligand interactions solely from structure remains a challenge. Recent studies have proposed various approaches to capture molecular surface patterns with functional relevance, such as 3D Zernike descriptors [41, 151] and geometric invariant fingerprint descriptors [149]. These approaches have limitations, as they rely on hand-crafted descriptors and manually optimized protein surface features. Therefore, determining the appropriate set of features for a given task becomes difficult.

In this work, we use our motion planning-inspired framework to extract the geometric features of the protein surface model. The work utilizes the critical points information to provide different conformations of the ligand molecule around the protein surface, which help in planning the feasible trajectory of the ligand to a protein's active binding site. Before going to the details of our approach, we will focus our discussion on the mathematical definitions of our algorithmic framework related to ligand and protein bio-molecules.

5.1.1 Algorithmic Design

5.1.1.1 Mathematical Concepts

We define the principal mathematical concepts, i.e., abstract simplicial complex, Vietoris-Rips complex, and discrete Morse theory w.r.t. the protein and ligand bio-molecules. The elements of the simplicial complex set S are called vertices, and these vertices refer to ligand conformations in the conformation space.

We perform steps discussed in Section 3.1 to generate a simplicial complex using Vietoris-Rips complex $R(S)$ to capture the topological structure of the protein surface, i.e., vertices, edges, triangles, etc. We apply the discrete Morse function, discussed in Section 3.2, on the same simplicial complex to extract the critical points information of the surface. The discrete setting of Morse theory avoids the overhead of differential topology, thus reducing the computation complexity for high dimensional structures.

We model the protein surface as a rigid object. Here, S is the set of all ligand conformations that connect to form a simplicial complex $R(S)$. These conformations are generated at a radial distance 2ϱ away from the surface to avoid collisions, such that $S \subseteq \mathcal{C}_{free}$. We

take ϱ as the diameter of the circumscribed circle of the ligand molecule. Considering the above parameters, we define the discrete Morse function as follows.

Definition 5.1 *(**Distance function in Conformation Space**): Let D be the Euclidean distance function that measures the distance between the point $x \in \mathcal{C}_{free}$ and the nearest point y on the protein surface P, i.e., $D(x) = \min_{y \in P} \|x - y\|$.*

Definition 5.2 *(**Density function in Conformation Space**): Let $\Gamma(y, \varrho)$ be a density function where $\varrho > 0$ and y is the point on the protein surface. The function Γ counts all neighbors close to y in S within distance ϱ.*

Definition 5.3 *(**discrete Morse function**): Let f be a discrete Morse function on $R(S)$ restricted to the vertices of the Vietoris-Rips complex. It is formally defined at any point in conformation space by*

$$f(x) = D(x) \times \Gamma(y, \varrho). \tag{5.1}$$

Please refer to Section 3.2 for our expanded definitions and theorems.

Definition 5.4 *(**Critical points**): The set of critical points is defined as the set of non-degenerate points on the surface of protein when the given discrete Morse function f reaches its extreme values, i.e., local minima or maxima.*

Definition 5.5 *(**Feasible critical points**): This set is defined as all possible ligand conformations in S at a radial distance of ϱ from a critical point on the protein surface. In other words, it is the union of intersections of vertices in S within the metric balls of radius ϱ centered at some critical point.*

5.1.1.2 Extraction of Protein surface's Geometric Features

Algorithm 7 describes how we construct a simplicial complex around the protein surface by sampling and connecting ligand conformations in method *ConstructComplex*. Using the sampling condition from [126], the algorithm performs simplicial collapse to remove redundant topological information, i.e., vertices and edges, and provides a 1-dimensional skeleton

of the simplicial complex around the protein surface in line 3, i.e., a surface mesh. It applies discrete Morse function f to this simplicial complex to identify the local maxima and minima curvatures on the protein surface model in line 4. The identified critical points of the surface structure are the highest and the lowest peak points at which function f reaches its extremum, i.e., protrusions and cavity. For function f, the distance becomes an equalizer, and the density becomes an essential factor affecting the Morse value at the surface curvatures.

Algorithm 7 Path planning to protein binding site

Input: P: Protein surface model, $Query$: query to be solved from start conformation to the binding site conformation of the ligand, R: Planned path to the binding site, H: a set of ligand conformations around the protein surface.

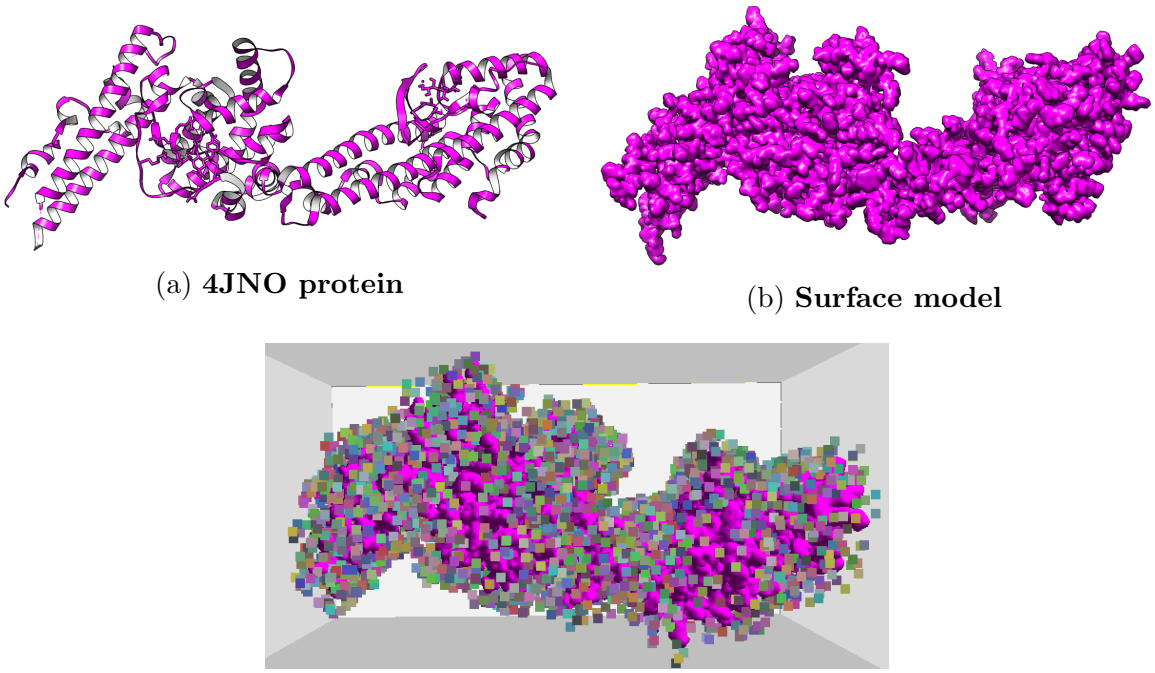
1: Let $R \leftarrow \{\phi\}$.
2: $S \leftarrow ConstructComplex(P)$; ◁ Refer Def. 3.1
3: $TopologicalCollapse(S)$; ◁ Refer Sec. 3.1
4: $C \leftarrow IdentifyCriticalPoints(S)$; ◁ Refer Def. 5.3, 5.4
5: $F \leftarrow GetFeasiblePoints(S, C)$; ◁ Refer Def. 5.5
6: $H = S \bigcap F$
7: $R = Query(H)$
8: **return** $\{H, R\}$

The algorithm extracts the feasible critical points at radial distance ϱ from the identified critical points of the protein surface in line 5. These conformations are determined near the protein surface and are part of the simplicial complex $R(S)$, refer to Def.7. It then uses the extracted geometric information map to plan a path for the ligand from the start conformation to the binding site conformation in lines 6-7. The output of our algorithm is an extracted geometric information map consisting of critical points, feasible critical points, and a trajectory from the start conformation to the binding site conformation.

5.1.2 Model Transformation

We obtain protein data from the Protein Data Bank (PDB) [18, 19] and construct their geometric structure using CHIMERA [97]. Figure 5.1 shows the graphical representation of 4JNO protein, its high-dimensional surface model, and the extracted geometric map.

We consider ten proteins and two ligand bio-molecules to study and understand the protein surfaces and their geometries. We use the high dimensional surface models of proteins



(a) **4JNO protein**

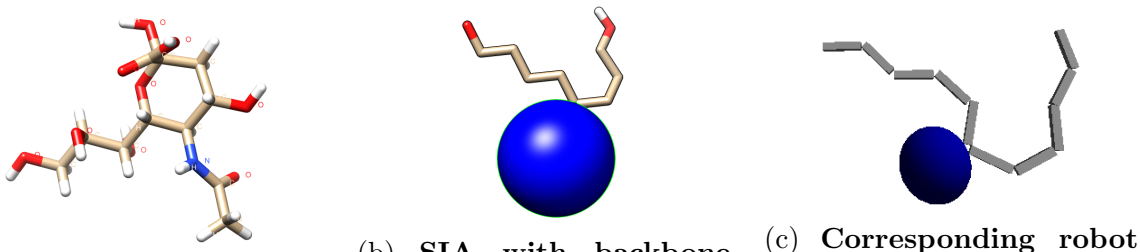
(b) **Surface model**

(c) **Geometric information map**

Figure 5.1: The figure shows the multiscale surface model of the 4JNO protein and the geometric features (critical points) detected on the protein surface. The geometric information map provides the point size view of the ligand molecule conformation around the surface.

as a stationary rigid body in the conformation space. The protein studied ranges from 36 to 1708 residues. We construct a flexible linkage model of the ligand using the covalent bond length and angle measurement derived from CHIMERA. Figure 5.2 shows the backbone structure of the SIA ligand molecule and our corresponding model. Each covalent bond simulates one link of the robot of length 1.53 $\mathring{A}$, and a sphere of radius 2.7 $\mathring{A}$ surrounds the part of the C-N bond of the molecule. Similarly, we perform the transformation for SO4 ligand where the covalent bond length is 1.47 $\mathring{A}$, and the angle between the pair of S-O bonds is 120° each, as shown in Figure 5.3.

We used 1SQ6 (225 residues), 1TQX (442 residues), 1ZRL (583 residues), 3NTJ (1544 residues), 4JNO (586 residues), 5JBE (1708 residues), 5ZT1 (669 residues), 6E02 (153 residues), 6JMI (375 residues), and 7OXS (36 residues) as protein models; and SO4 and SIA as ligand molecules for the analysis. The proteins selected include three *Plasmodium Falciparum (PF)* pathogen proteins, i.e., 1SQ6, 1TQX, and 3NTJ, and one DNA protein (7OXS). *PF* inflicts the most damage and is responsible for many malaria-related deaths. The high mutational capacity, coupled with the changing metabolism of the pathogen, makes



(a) **Ligand SIA** (b) **SIA with backbone structure** (c) **Corresponding robot model**

Figure 5.2: Robot transformation of SIA ligand. The blue sphere encloses the C-N bond part of the ligand.

malaria drug treatment development an evolving problem.

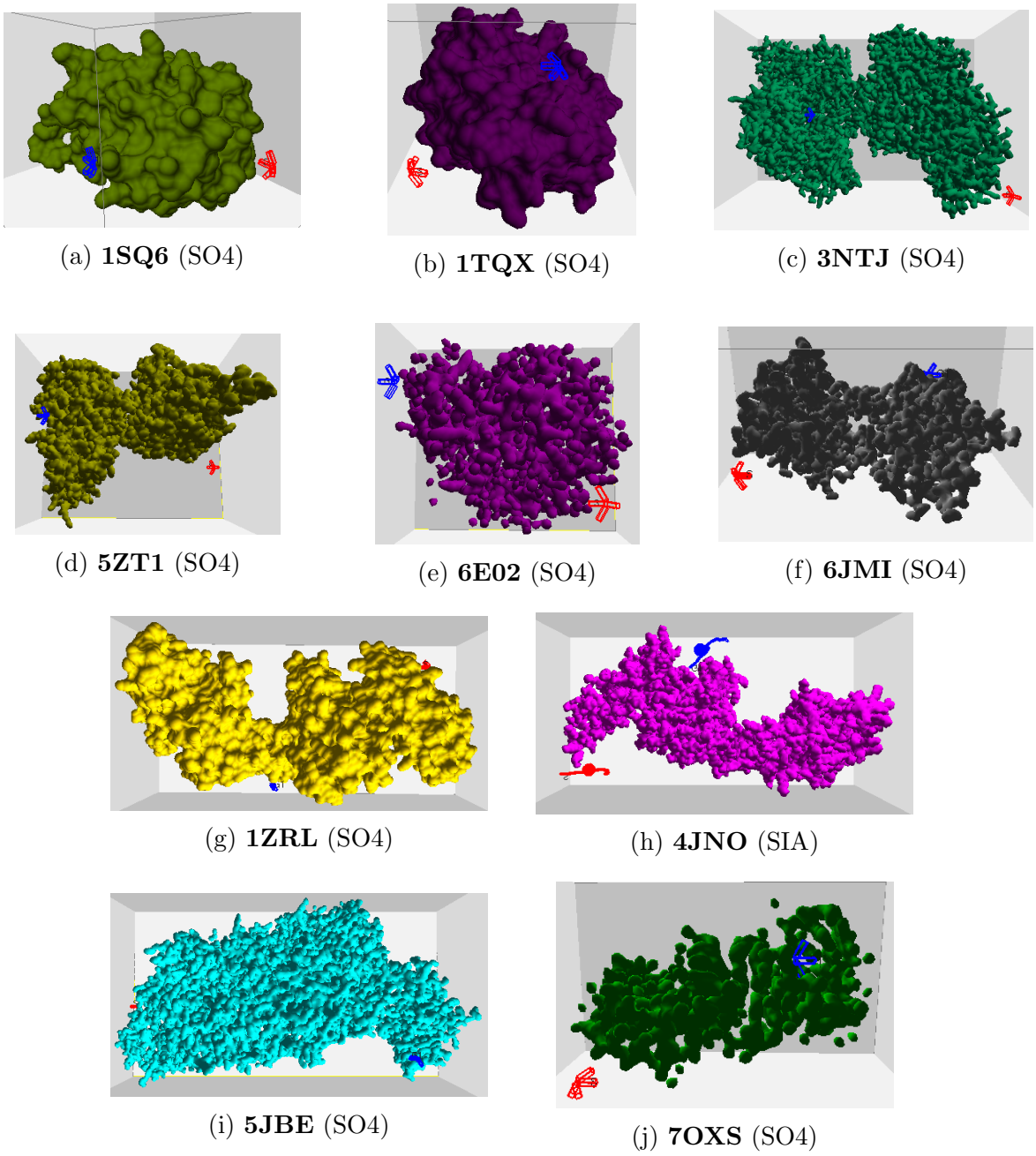
(a) **Ligand SIA** (b) **Corresponding robot model**

Figure 5.3: SO4 ligand into a robot transformation.

We follow the same steps of transformation for all protein and ligand bio-molecules to avoid the loss of biological significance. We transform the coordinates of the known binding site provided in the PDB file of each protein into the goal conformation for our ligand model in the conformation space. The start and goal positions are highlighted in red and blue, as shown in Figure 5.4. The caption of each protein surface model represents the ligand molecule used for the protein-ligand interaction experiment. In particular, protein 4JNO binds with ligand SIA, and the rest binds with ligand SO4.

5.1.3 Experimental Analysis

We perform experiments in 10 different protein conformation spaces and average the results over five random trials for geometric map generation and five random trials for trajectory planning time in each protein's conformation space.



(a) **1SQ6** (SO4) (b) **1TQX** (SO4) (c) **3NTJ** (SO4)

(d) **5ZT1** (SO4) (e) **6E02** (SO4) (f) **6JMI** (SO4)

(g) **1ZRL** (SO4) (h) **4JNO** (SIA)

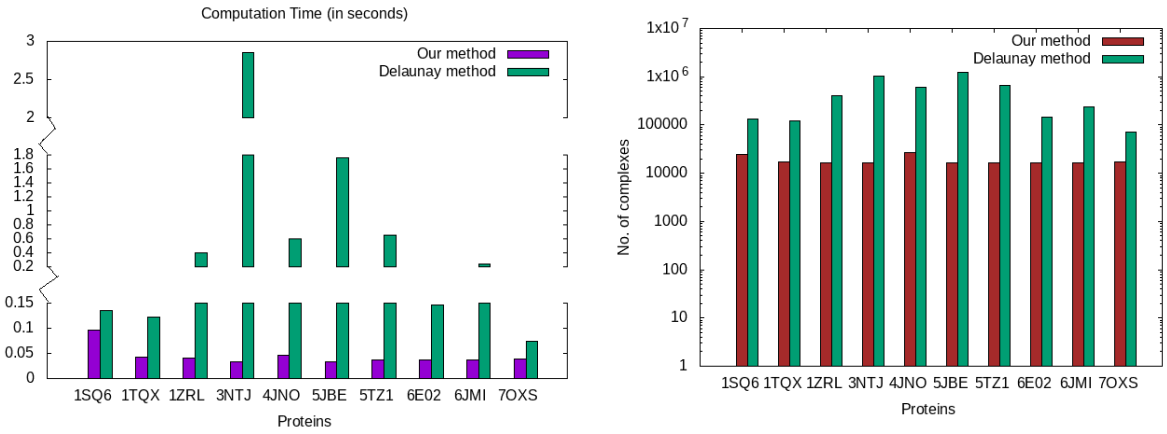
(i) **5JBE** (SO4) (j) **7OXS** (SO4)

Figure 5.4: Protein surface models studied

5.1.3.1 Computing surface complexes

We compare the performance of our method with the Delaunay-refinement-based method from the TetGen library due to its relevance to our approach, as seen in [54]. We record the computation time and total complexes generated to capture the topological structure of the protein surface, i.e., a surface mesh.

In Figure 5.5, we observe that the Delaunay method requires a longer duration to generate complexes for all proteins than our strategy. The number of complexes generated by the Delaunay method for proteins 3NTJ and 5JBE are equivalently highest among all proteins, but the computation time difference between them is significant and inconsistent. On the other hand, to maintain the skeleton structure of the simplicial complex, our method shows a consistent relation between the computation time and the total generated complexes. The calculation performed to construct complexes around the surface does not affect the overall performance resulting in less memory overhead. Thus, our method is more reliable and faster in capturing the topological structure of the protein surface by generating fewer complexes in low computation time than the baseline method.



(a) **Total computation time (in seconds)**

(b) **Total complexes generated**

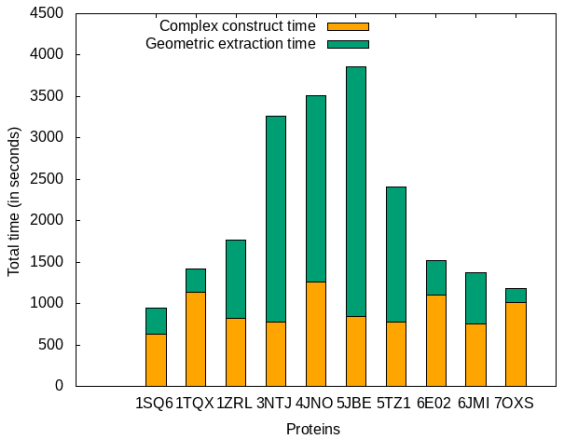
Figure 5.5: Qualitative performance analysis of generated surface complex.

On the generated surface complex, the work [54] performed numerical calculations to identify the surface curvatures. Instead, our method provides an automated framework for determining the minimum and maximum curvatures of the protein surfaces, i.e., critical points, and generates feasible critical points around them, as discussed next.

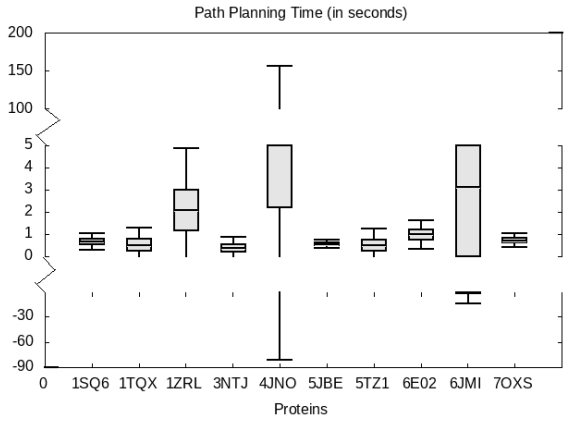
5.1.3.2 Motion planning towards a binding site

Our method generates a geometric information map for all proteins, similar to that shown in Figure 5.1c. We input the map to PRM [68] method as an initial graph to solve the query from start conformation to the binding site conformation. We show the distribution of time taken to construct a simplicial complex, compute a geometric map, and plan a path

to the binding site. Figure 5.6a shows the total computation time taken by our method to capture protein surface topological and geometric information. And Figure 5.6b provides the time consumed to plan a successful path to the goal conformation over five trials. We observe that for large protein structures, i.e., 1ZRL, 3NTJ, 4JNO, 5JBE, and 5ZT1, the computation of geometric information is higher than the complex construction time. On the other hand, the ligand conformations generated around the identified curvatures of protein surfaces contribute to the smooth navigation of the ligand to the goal conformation at a safe distance from the protein. The path planning time for all proteins becomes negligible, and the highest planning time recorded was 150 seconds for the SIA ligand around the 4JNO protein surface.



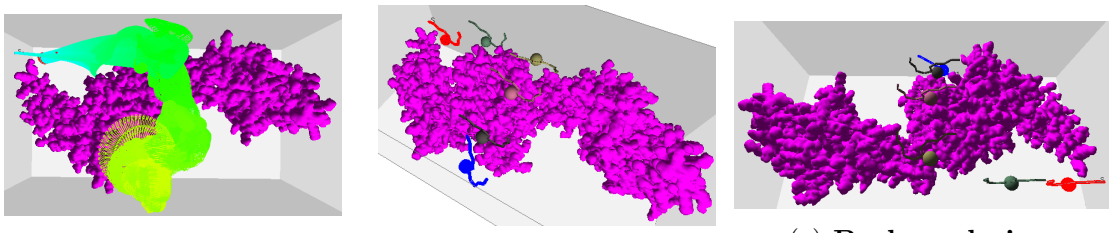
(a) **Total time taken to construct the simplicial complex and extract geometric features of the protein surface.**



(b) **Path planning time range shown over mean and standard deviation values for all proteins.**

Figure 5.6: Quantitative time analysis of our algorithm.

Figure 5.7 shows screenshots of the planned pathway for the ligand SIA around the 4JNO protein surface to the goal conformation. The different view angles reflect the motion of the ligand biomolecule around the protein surface using the ligand conformation generated by our method. Hence, we noticed that our algorithm could capture the biological aspect of the protein-ligand interaction using the geometric information map. We produce the same results for the remaining nine proteins.



(a) **Top view of the Path** (b) **Side view** (c) **Backward view**

Figure 5.7: Trajectory planned using feasible critical points information (ligand conformations generated around 4JNO protein surface) to the binding site. The side and backward views show intermediate ligand conformations of the path.

5.1.3.3 Binding affinity of goal conformation

We are interested in analyzing the relevance of our goal conformation with known association sites for the ligand. We validate the identified ligand conformations for each protein with the native binding pose using the binding affinity measure. We use the molar Gibbs free energy ΔG (binding affinity) to determine the relevance for the binding pose. Gibbs free energy is a thermodynamic potential that measures the capacity of a thermodynamic system to do maximum or reversible work at a constant temperature and pressure (isothermal, isobaric) [59]. The protein-ligand binding occurs only when the change in Gibbs free energy ΔG of the system is negative, i.e., when the system reaches an equilibrium state at constant pressure and temperature. Table 5.1 shows the ΔG value for the ligand at our predicted goal conformation compared with the known ΔG value at the native binding site for all ten proteins.

Table 5.1: Binding Affinity (Kcal/mol) for the ligand at goal conformation compared to ligand native pose for each protein.

	1SQ6	1TQX	1ZRL	3NTJ	4JNO	5JBE	5ZT1	6E02	6JMI	7OXS
Our method	-5.6	-5.2	-5.3	-5.4	-7.0	-5.2	-5.2	-5.2	-5.2	-5.2
Native poses	-5.4	-5.4	-5.2	-5.2	-7.0	-5.2	-5.3	-5.2	-5.2	-5.7

We observe that our method provides a closely relevant binding affinity for our ligand goal conformation compared with the native binding poses of the ligand for each protein. Overall, we conclude that our method successfully captures the geometric features of the protein surfaces and plans a path for ligand biomolecule to the binding site without losing its biological significance.

5.2 Protein-Protein Interactions

This work studies the binding behavior between a structured protein and a disordered/misfolded protein during protein-protein interaction. The disordered proteins are also known as Intrinsically Disordered Proteins (IDPs). IDPs do not have distinct, well-defined secondary and tertiary structures because of their remarkable backbone flexibility [142]. When an IDP binds to a macromolecule (usually another protein), large interfaces get involved, resulting in specific but comparatively weak interactions. IDPs common in genomes and proteomes of a living organism have many occurrences in eukaryote groups. They are prevalent in various human diseases and enriched in cardiovascular disease, diabetes, cancer, and neuro-degenerative disease-related proteins [73]. The disordered region can happen spontaneously because millions of copies of proteins get generated during the lifetime of a living organism, making humans become an easy target for many infectious diseases, precisely through host-pathogen interactions [35, 114].

Pathogen like *Plasmodium Falciparum* (PF) is a protozoan parasite of humans that inflicts damage to the human immune system and is responsible for most malaria-related deaths [63]. Plasmodium infection of mammals begins with the injection of the sporozoite into the skin of the vertebrate host during the bite of a female Anopheles mosquito. This results in growth and multiplication first in the liver cells and then in the red blood cells leading to kidney failure, severe anemia, and many more [154]. We consider host-pathogen interaction between PF pathogen and the human/mice IDPs to study and analyze the binding behavior of IDPs in structure-based molecular interactions.

The intrinsic disorder poses a challenge for both experimental analyses of the conformation and computational modeling due to the lack of stable structure. Despite the instability, it is critical to understand the biological functionality during protein-protein interactions. To mitigate this, we propose a topology-based rigid-body docking algorithm that takes the protein surface models of the globular proteins to predict a binding conformation for the interacting IDPs. Our approach extracts the topological and geometric properties of the protein surface to generate random IDP conformation ensembles around it. It then ranks the conformation ensembles based on the docking score to find the geometrically favorable pose. The algorithm examines the score values to select the geometrically-favorable binding position and plans a feasible trajectory from IDP's initial location to it. Our method can

be used, as a tool, to find the best docking position that is geometrically fitting on the protein surface model when no information other than the individual structures is available. Figure 5.8 shows an overview of our algorithm.

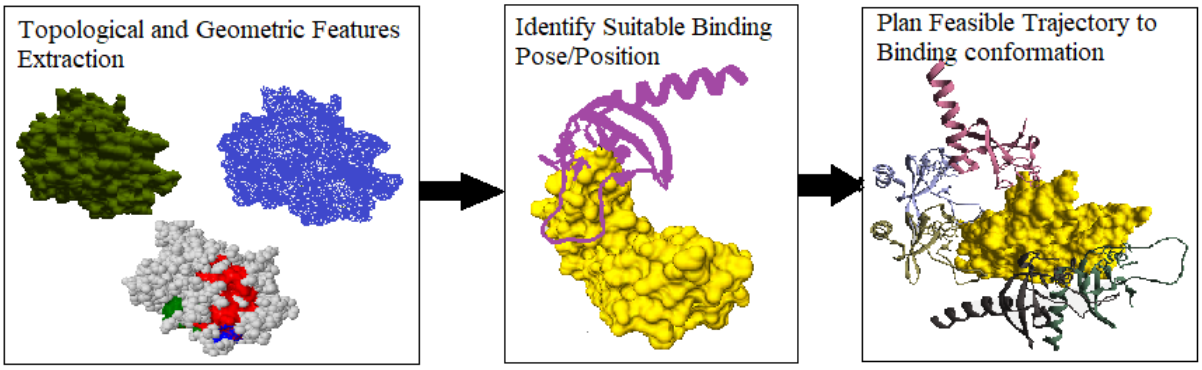


Figure 5.8: Workflow of our approach.

5.2.1 Method

We use the similar mathematical definitions discussed in Section 5.1.1.1 where the vertices of simplicial complex S are IDP conformations instead. We apply our algorithmic framework to predict the geometrically-favorable binding pose for IDPs around the studied structured/globular proteins.

5.2.1.1 Finding a suitable docking conformation

Algorithm 8 constructs a simplicial complex around the protein surface by sampling and connecting IDP conformations in method *ConstructComplex*. On satisfying the sampling condition from [126], the algorithm performs simplicial collapse to remove redundant topological information, i.e., vertices and edges, and provides a skeleton of the simplicial complex around the protein surface in line 3, i.e., a surface mesh. Recall that we refer to vertices as the IDP conformations, and the edges are the lines that connect the to/fro movements of IDP between two conformations. It applies discrete Morse function f from [124] to this simplicial complex to identify the local maxima (protrusions) and minima (cavity) curvatures of the protein surface, i.e., critical points, in line 4. The identified critical points are the highest and the lowest peak points on the surface at which function f reaches its extremum.

Algorithm 8 Sampling and planning path to binding pose

Input: P: Protein surface model, R: A planned pathway to the binding site, s: initial IDP conformation, H: set of closest IDP conformations around the protein surface, g: best binding pose.

1: Let $R \leftarrow \{\phi\}$.
2: $S \leftarrow ConstructComplex(P)$; ◁ Refer Def. 3.1
3: $TopologicalCollapse(S)$; ◁ Refer Sec. 3.1
4: $C \leftarrow IdentifyCriticalPoints(S)$; ◁ Refer Def. 5.3, 5.4
5: $F \leftarrow GetFeasiblePoints(S, C)$; ◁ Refer Def. 5.5
6: **for all** x ∈ F **do**
7: **for all** $c \in C$ **do**
8: **if** x closest to c **then**
9: $H[x] = d_{pose}(x, c)$ ◁ Refer Eq. 5.2
10: $g = \forall_{x \in H} min(H)$
11: $R = PlanPath(s, g)$ ◁ Refer Sec. 3.3
12: **return** $\{S \bigcap F, R\}$

The algorithm then extracts the feasible critical points at radial distance ϱ from the identified critical points of the protein surface in line 5. These feasible critical points are the conformations close to the protein surface and are part of the simplicial complex $R(S)$, refer to Def. 5.5 in Section 5.1.1.1. Next, we consider the closest conformations as the set of predicted conformations for an IDP and use it to evaluate and determine the ranks of the conformations using Eq. 5.2. From the predicted conformations, a geometrically favorable binding position of the IDP gets selected such that the conformation is closest to protein surface curvature in lines 9-14. We use the Hausdorff distance to measure the distance between the protein surface (P) and the IDP conformation (I) to find the geometrically suitable docking position, as discussed below.

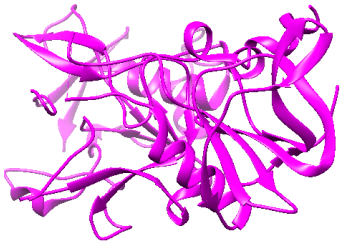
$$d_{pose}(P, I) = \max\{\sup_{p \in P} \inf_{i \in I} d(p, i), \sup_{i \in I} \inf_{p \in P} d(p, i)\}. \tag{5.2}$$

It takes the conformation with the minimum Hausdorff distance as the final docking position from all the possibly generated conformations. Finally, a path is planned for the IDP from the start conformation to the binding pose conformation taking the other predicted IDP conformations as waypoints (line 14). The process of selecting a binding pose happens internally, where our method ranks and automatically chooses the binding conformation to plan the path during the interaction of protein-protein complexes. As a result, our algorithm outputs an extracted geometric information map consisting of critical points, feasible critical

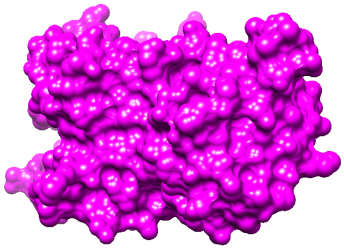
points (predicted IDP conformations), and a pathway from the start conformation to the binding pose conformation.

5.2.2 Experimental Data

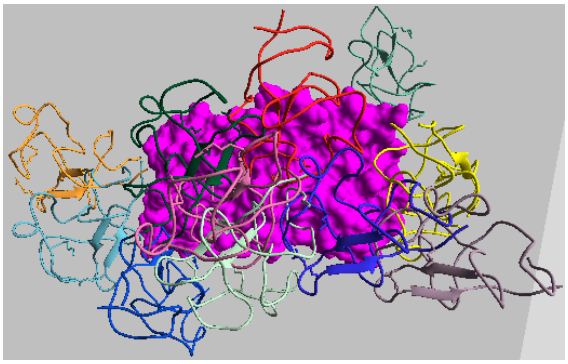
We obtain protein data from the Protein Data Bank (PDB) [18, 19] and construct their tertiary structure using CHIMERA [97]. We obtain IDP data from PDB and AlphaFold Protein Structure Database (AlphaFold DB) [129]. We consider nine proteins and six IDP bio-molecules to study and understand the biological binding mechanism of IDPs using protein surface geometries. The high-dimensional surface models of proteins represent a stationary rigid body in the conformation space. Figure 5.9 shows the tertiary-structure representation of 3SRI protein, its high-dimensional surface model, and the IDP conformation ensembles around it.



(a) **3SRI protein**



(b) **Surface model**

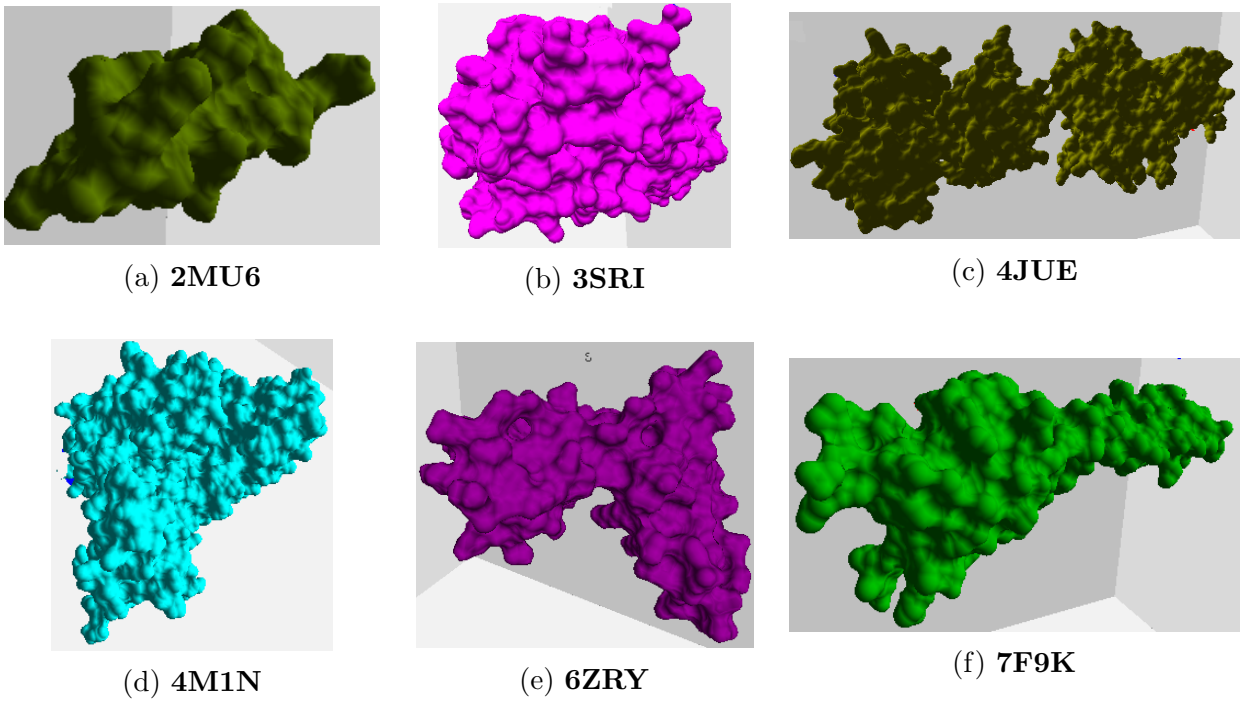


(c) **IDP conformation ensembles**

Figure 5.9: The figure shows the multiscale surface model of the 3SRI protein and the predicted IDP conformations around detected geometric features (critical points) of the protein surface. The conformations viewed in (c) are the top 10 predicted 1KRN bio-molecule conformations around the surface model.

The proteins selected include nine *Plasmodium Falciparum (PF)* pathogen proteins,

i.e., 1SQ6, 1TQX, 2MU6, 3NTJ, 3SRI, 4JUE, 4M1N, 6ZRY and 7F9K, as shown in Figures 5.10 and 5.11. *PF* is responsible for most malaria-related deaths and forms part of our ongoing research into identifying feasible protein drug targets. The high mutational capacity, coupled with the changing metabolism of the pathogen, makes the development of malaria drug treatments an evolving problem. In this work, we are interested in studying and analyzing the behavior of PF pathogens in the PPI network. Hence, these proteins were selected as they are the potential targets for malaria inflicts.



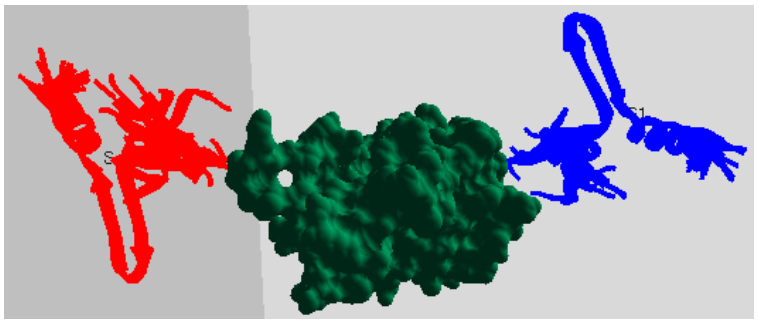
(a) **2MU6** (b) **3SRI** (c) **4JUE**

(d) **4M1N** (e) **6ZRY** (f) **7F9K**

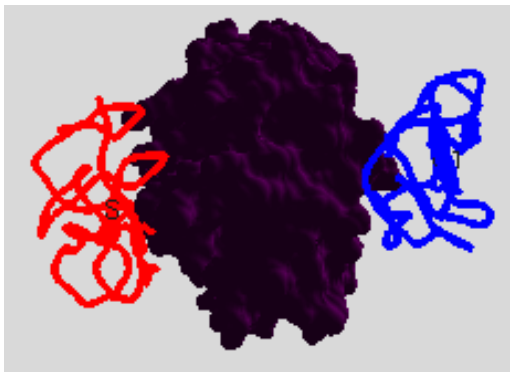
Figure 5.10: The figure shows the tertiary structure of PF pathogen proteins taken into consideration for the experiment analysis.

We selected 1KRN (88 residues), 2LE3 (42 residues), 5EJW (91 residues), and 7KPI (142 residues) proteins as IDP based on their high disorder behavior shown in the protein feature view plot available on the PDB database. The other IDP bio-molecules, AF-I1E4Y1-F1 (117 residues) and AF-P59773-F1 (190 residues), from AlphaFold DB, are of mus-musculus and homo sapiens species, respectively. The mean per-residue confidence score (pLDDT) for AF-I1E4Y1-F1 is 48, and for AF-P59773-F1 is 59. The pLDDT measure estimates whether the predicted residue has similar distances to neighboring C-α atoms (within 15 $\mathring{A}$) in agreement with the naive structure and scored between 0 and 100. The score assesses the local

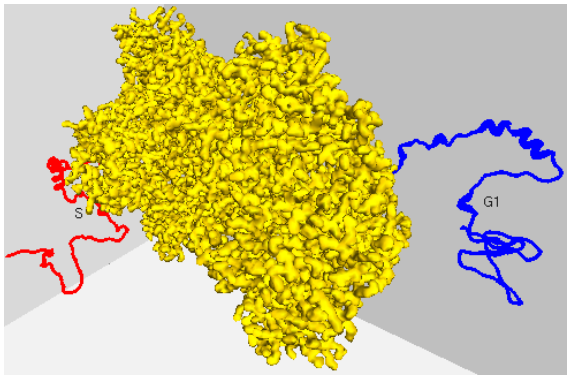
model quality of the structure, i.e., a lower score refers to the existence of more disordered regions in a bio-molecule. The selected IDPs are bio-molecules of humans and mice susceptible to malaria.



(a) **1SQ6** (2LE3)



(b) **1TQX** (1KRN)



(c) **3NTJ** (AF-I1E4Y1-F1)

Figure 5.11: The figure captures a random combination of a globular protein surface model and an IDP from the experimental analysis, with IDP names mentioned in the brackets. The red color conformation refers to the start position, and the docking position is in blue.

Figure 5.10 shows the surface model of six PF pathogen proteins, and Figure 5.11 shows a random combination of IDPs interacting with the remaining three proteins in their start (red) and goal (blue) positions. We conduct tests on every PF protein complex with all 6 IDPs, resulting in a total combination of 54 protein-protein complexes.

5.2.3 Result Evaluations

We evaluate performance using quantitative and qualitative measures for all IDPs with each PF protein for geometric feature extraction, path planning to dock position, and binding affinity measure. Overall we executed 1250 experiments and averaged the result

values over ten runs. We compare our method's performance with two baseline methods, i.e., HawkDock [136] and HDOCK [145].

5.2.3.1 Quantitative Analysis

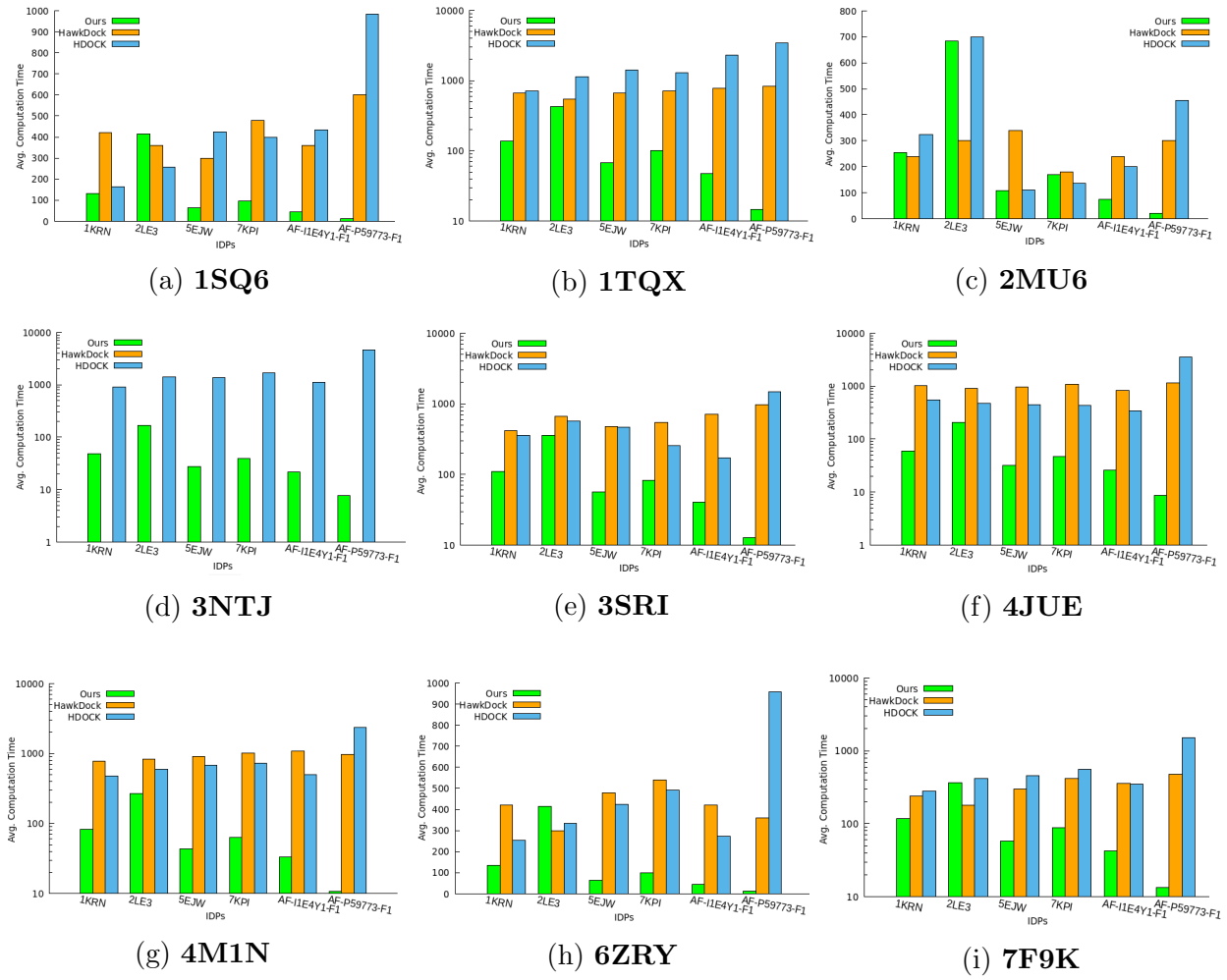
Extracting geometric features of the protein surface: Recall that our method constructs a surface mesh (or simplicial complex) around the considered protein surface models to abstract the topological and geometric information. During the execution, it randomly samples the IDP conformations around the protein surface model, thus, constructing a manifold mesh representation to capture the topology of the protein's surface. These topological features aid in identifying the geometric properties of the protein surface, i.e., minima or maxima, for better approximation, as shown in Figure 5.8. Next, our method uses geometric information to find possible binding conformations around the protein surface and apply the scoring function from equation 5.2 to get the top 10 geometrically-fitting docking conformations for an IDP. Figure 5.1c shows the top 10 predicted association conformations of 1KRN IDP around the 3SRI protein surface model. We observed that the feature extraction process is independent of the globular protein's size and has minimal effect on the performance of our algorithm, making it suitable for macro-molecules, as discussed next.

Computational time: We analyze the computation time (in seconds) required for feature extraction and prediction of geometrically favorable docking conformations in these IDP-protein interactions. This study includes all IDPs in nine PF pathogen protein conformation spaces. The feature extraction time measures the duration of extracting topological and geometric features from the globular protein surface, while the ranking time finds the top 10 conformations. To assess our algorithm's performance efficiency, we compare our total time to output the top 10 docking conformations with HawkDock and HDOCK, as depicted in Figure 5.12.

Our method demonstrates the faster prediction of IDP binding conformations compared to HawkDock and HDOCK in all PF pathogen protein conformation spaces except for 2LE3. The smaller size of the 2LE3 IDP leads to a longer conformation sampling time necessary for accurate feature capture in large or complex-size proteins. Proteins like 1SQ6, 6ZRY, and 7F9K have tetrahedral polygonal shapes rather than spherical surface structures, thus,

allowing the sampling of conformation ensembles in various fitting positions. As a result, it takes extra time in the case of the 2LE3 IDP within the conformation space of these proteins. However, this difference does not affect our method's performance significantly and results in less time overhead compared to the baseline methods. Figure 5.12 highlights that our method outperforms the baseline methods in most protein conformation spaces, despite minimal time overhead.

In particular, HawkDock fails to find docking conformations for the 3NTJ protein due to its limitation to proteins with fewer than 1000 amino acids.



(a) **1SQ6** (b) **1TQX** (c) **2MU6**

(d) **3NTJ** (e) **3SRI** (f) **4JUE**

(g) **4M1N** (h) **6ZRY** (i) **7F9K**

Figure 5.12: The plots show the total computation time taken (in seconds) by all three methods to predict the top 10 IDP docking conformation ensembles around the protein surface model.

We analyzed that using the geometric information of protein surface, it is still possible

to predict multiple structural arrangements of IDPs around the proteins to find the closest interacting binding pose between two bio-molecules without declining computation performance. Thus, we can conclude that the amount of data assessed by our method does not impact its surface approximation and still provides a quantitatively good performance.

5.2.3.2 Qualitative Analysis

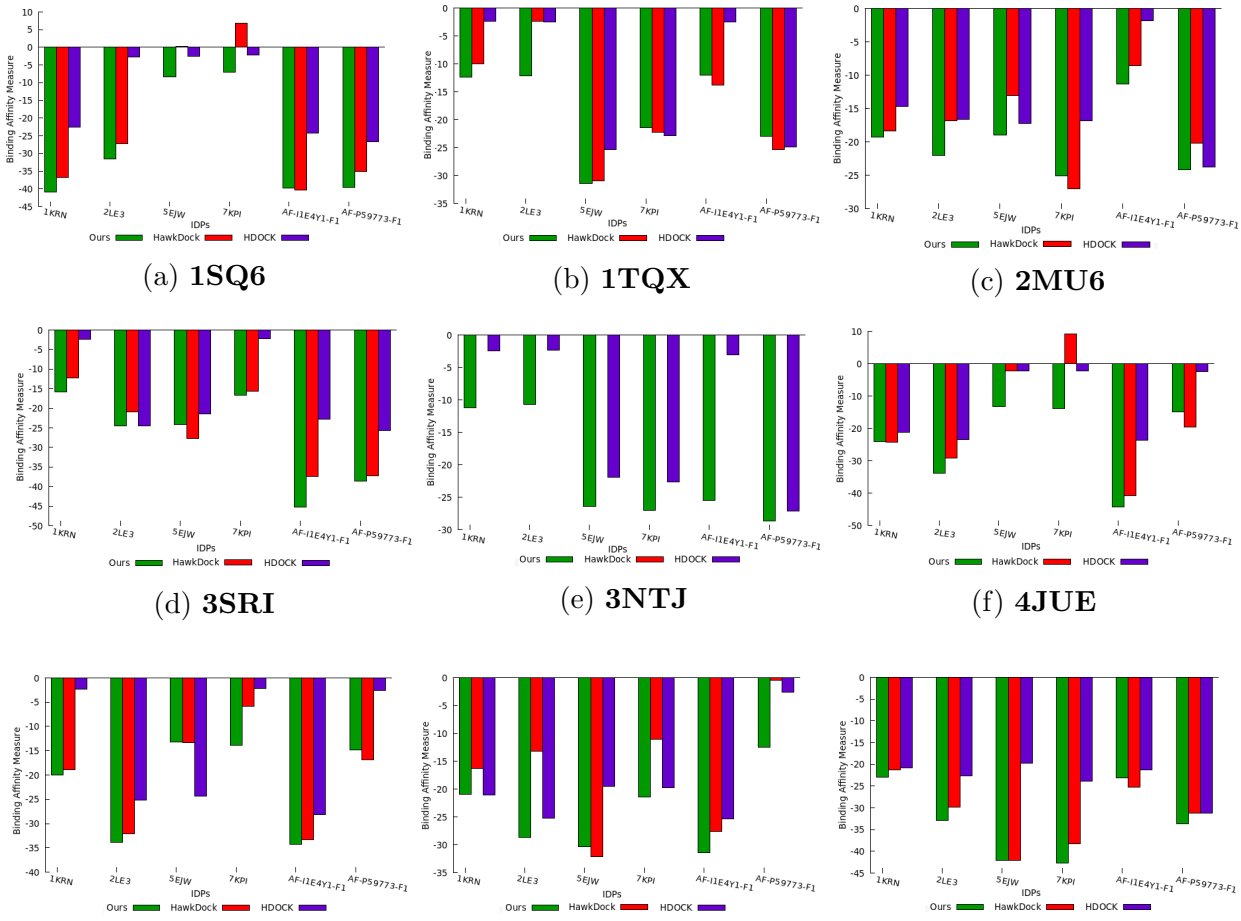
Selecting the suitable binding conformation: As initially mentioned, our method predicts the top 10 docking conformations for an IDP across all PF pathogen protein conformation spaces. This process iterates over ten times, and for each iteration, we record the top 10 conformations to assess the likelihood of obtaining the same conformation from ten random iterations. The recorded outputs are then further analyzed to identify the IDP conformation with the highest frequency as the most suitable docking position among the ten experimental runs. This selected conformation is then subsequently utilized as input for path planning. In Figure 5.11, examples of best binding poses (goal positions) for 3 IDPs are depicted in blue. To validate the quality of the chosen binding pose for protein-protein interactions, we examine the binding affinity before proceeding with path planning, as elaborated in the subsequent discussion.

Binding affinity measure: We compare the binding affinity of our IDP binding conformation with the binding affinity computed for the IDP conformations predicted by HawkDock and HDOCK methods across all PF pathogen proteins. The molar Gibbs free energy ΔG is used to assess the relevance of the binding pose. Gibbs free energy is a thermodynamic potential that quantifies the maximum reversible work capacity of a thermodynamic system under constant temperature and pressure (isothermal, isobaric) conditions [59]. Protein binding occurs when the change in Gibbs free energy ΔG is negative, indicating equilibrium at constant pressure and temperature.

We utilize the molar Gibbs free energy ΔG to calculate the binding affinity of the top-ranked IDP conformation ensemble predicted by all three methods. Figure 5.13 illustrates the binding affinity measure of our predicted IDP binding pose for each protein compared to the binding affinity measures obtained for the IDP conformations predicted by HawkDock and HDOCK methods. HawkDock exhibits a positive binding affinity for the 7KPI

IDP conformation when interacting with 1SQ6 and 4JUE proteins. In contrast, our algorithm consistently predicts IDP conformations with negative binding affinities for all IDPs interacting with PF pathogen protein complexes. This evidence indicates a stronger association displayed by our geometrically-favorable docking positions and consistency achieved in identifying favorable binding conformations through our method.

As mentioned previously, HawkDock failed to predict the docking conformation for the 3NTJ protein, resulting in the absence of a binding affinity measure for this case.



(a) **1SQ6** (b) **1TQX** (c) **2MU6**

(d) **3SRI** (e) **3NTJ** (f) **4JUE**

(g) **4M1N** (h) **6ZRY** (i) **7F9K**

Figure 5.13: The plot shows the Binding Affinity measure for the top-most IDP docking conformation predicted by the three methods.

Based on the observations in Figure 5.13, we consistently find that our predicted docking conformations exhibit a negative binding affinity for all IDPs, surpassing the binding affinity of the IDP conformation ensemble generated by the baseline methods. Additionally,

we deduce that our method performs well even for macro-molecule proteins, such as 3NTJ, surpassing HDOCK and not being limited to small bio-molecules. Overall, our experimental conformations demonstrate better binding affinity in 95% of the compared cases, highlighting the significance of utilizing protein surface model features in generating conformations with favorable binding affinity outcomes. Consequently, we can conclude that the quality of our binding conformations competes favorably with the binding conformations predicted by existing approaches utilizing coarse-grained force field docking (HawkDock) and knowledge-based template-free docking (HDOCK).

Affinity comparison for known IDPs: We analyzed the binding affinity of protein-protein complexes specifically by focusing on folding-upon-binding [106]. To validate our method-predicted docking conformation, we compared our results with the work done in [11, 37, 43], which studied the interaction mechanism of IDPs in terms of structure, dynamics, affinity, and kinetics. To examine our results, we considered the same protein complex compounds as those studied in the aforementioned work, i.e., 4HTP, 3W1G, 3ALO, and 1SB0. Table 5.2 displays the binding affinity of the known and our predicted docking conformations for these protein complexes. Our findings revealed that our geometrically-fitting binding conformation exhibited a closely similar binding affinity to the known binding affinity of these protein complexes.

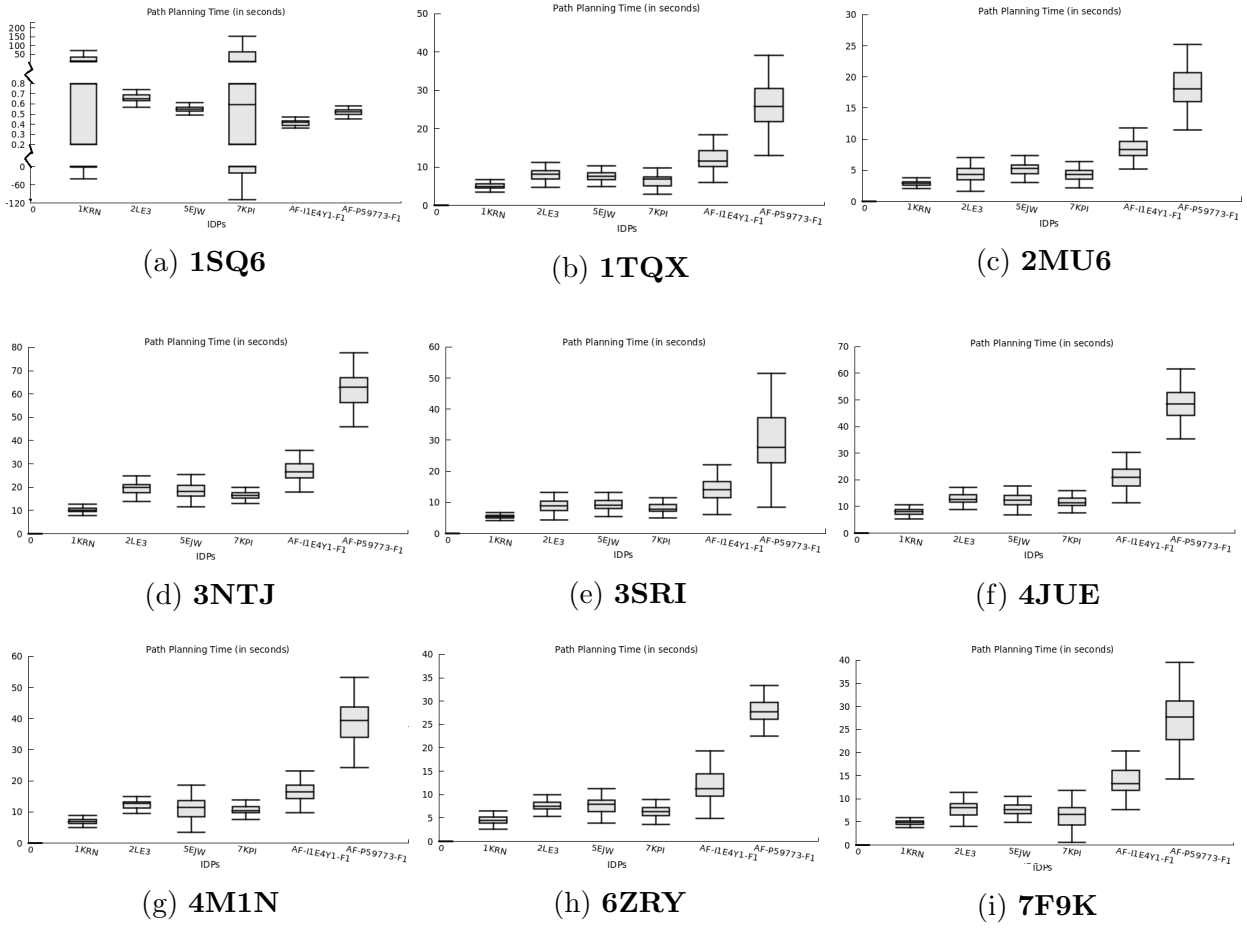
Table 5.2: Binding Affinity comparison for known IDPs

Protein complex	**IDP**	**PDB compound**	ΔG_{known}	ΔG_{pred}
$Artemis^{457-502}$	DNA Ligase-IV	4HTP	-7.7	-8.1
$Artemis^{593-621}$	DNA Ligase-IV	3W1G	-6.9	-5.9
p38 peptide	MKK4	3ALO	-3.7	-5.11
KIX domain of CBP	c-Myb	1SB0	-7.3	-7.5

5.2.3.3 Path planning to geometrically-favorable binding position

In addition to predicting binding conformations for rigid-body docking, our method also includes feasible trajectory planning toward the selected finalized binding pose during re-scoring. We assess the total time required for path planning to the predicted binding pose for all IDPs in the nine globular protein conformation spaces, as presented in Figure 5.14.

The path planning time represents the duration incurred for an IDP to transition from its initial conformation to the binding conformation while moving closer to the protein surface.



(a) **1SQ6** (b) **1TQX** (c) **2MU6**

(d) **3NTJ** (e) **3SRI** (f) **4JUE**

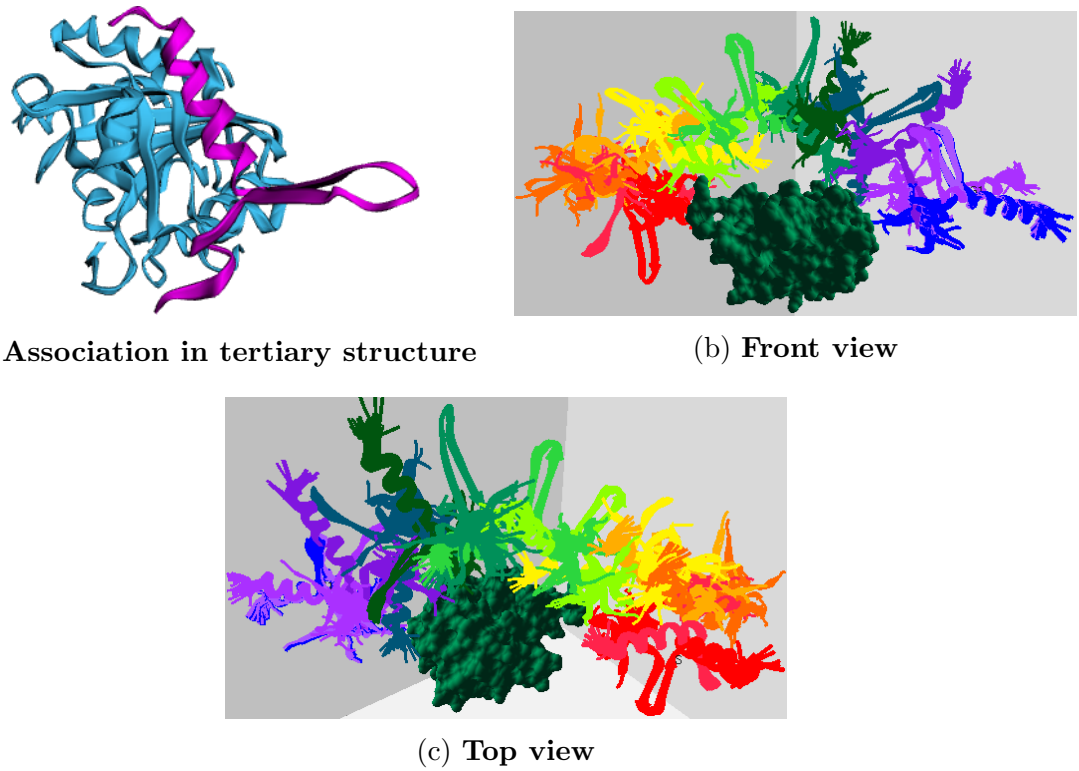
(g) **4M1N** (h) **6ZRY** (i) **7F9K**

Figure 5.14: The total time taken (in seconds) to plan a path for all IDPs in each protein's conformation space.

Figure 5.14 illustrates the distribution of path planning times for all IDPs across different protein conformation spaces. The y-axis represents the averaged path planning time over ten runs, while the x-axis represents the IDPs interacting with the respective proteins. The plot showcases the variability in path planning time, depicting the duration needed to move IDPs from their initial positions to the docking positions around the protein surface. In several protein conformation spaces, the difference between the minimum and maximum planning times is small or negligible for IDPs exhibiting a lower deviation, indicating that the planner consistently finds a similar route majority of times out of the ten runs. However, the 1KRN and 7KPI IDPs in the 1SQ6 protein's conformation space take longer time

spans. This behavior can be attributed to the broader structure of these IDPs, affecting their movement near the 1SQ6 protein surface and resulting in varying time values. Among all the studied IDPs, AF-P59773-F1 demonstrates the vast structure and highest disordered regions, making it challenging to plan its path while considering its structural transformations. Thus, we deduce from Figure 5.14 that the path planning period for the AF-P59773-F1 IDP is generally higher than other IDPs in most protein conformation spaces.

The unpredictable behavior of IDPs around the studied proteins enables us to analyze the feasibility of their interaction with specific proteins, particularly how easy they align around a protein structure for association. Path planning time provides insights into the locomotion of IDPs around proteins as they search for the most suitable binding pose for rigid-body docking. When used in conjunction with other tools to examine the conformational flexibility of IDPs during their motion around proteins can simplify flexible docking tasks by focusing computational methods solely on the dynamic structure of IDP conformations as they traverse the planned trajectory, facilitating future biological studies.



(a) **Association in tertiary structure**

(b) **Front view**

(c) **Top view**

Figure 5.15: The figures display a path planned for 2LE3 IDP around the 1SQ6 protein surface model using the geometrically favorable conformation ensembles. The start conformation is in red, and the binding goal position is in dark blue.

Figure 5.15 displays screenshots of the planned path for the 2LE3 IDP around the 1SQ6 protein surface, depicting the motion of the IDP biomolecule from its initial position to the experimentally predicted binding pose conformation. Different view angles illustrate the IDP's movement around the protein surface, and the intermediate conformations represent the IDP conformations generated during feature extraction, serving as waypoints. These intermediate conformations between the starting and goal positions demonstrate the structural transformations of the 2LE3 IDP as it moves in the vicinity of the 1SQ6 protein surface. Similar movements and structural arrangements occur for remaining IDPs across different protein conformation spaces.

We conclude that our approach successfully captures the geometric features of the protein surfaces and plans a path for IDP bio-molecule to the geometrically favorable binding pose showing a higher affinity compared to affinity measures by baseline methods. Thus, our results show the significance of our topology approach for future biological studies.

CHAPTER 6

Conclusion

This work presents a novel framework that aims to extract topological and geometric properties of the configuration space or a protein surface model. Our approach leverages the application of Topological Data Analysis methods such as Vietoris-Rips complex and discrete Morse theory to capture the properties of the underlying space in low computation time and less memory usage. We demonstrated that our framework applies to various sampling-based planners and complex robot systems. Through this research, we saw the immense benefit of these properties. Our methodologies have revealed the impacts of these features in multiple domains, including robotics and structural biology.

Specifically, we designed, developed, and evaluated the heterogeneity of the planning space to motivate further study into topology-aware sampling-based approaches. In Chapter 3, we introduced the mathematical definitions of topology, i.e., topological and geometric features, and the tools used to capture this information. We provided a theoretical foundation for our algorithm and experimentally showed its significance in performance improvement. We showed the usage of our framework for graph-based and tree-based methods and enhanced our framework into an incremental batch-based approach. Finally, we showed that our framework has various interesting applications in motion planning problems (Chapter 4). Firstly, the knowledge of critical points can help classify and identify different paths in high-dimensional spaces. Secondly, the ranked distinct routes can solve the MPV problem for fault-tolerant path planning of articulated linkage robots. Additionally, we demonstrated an application of our framework in computational biology (Chapter 5), where extracting the geometric features of a biomolecule's topological surface structure can assist in finding a geometrically-fitting pose for an interacting bio-molecule.

To further explore the beneficial use of the environment's properties, we believe that combining topology roadmap and machine learning techniques could be a game-changer. One idea that we're interested in investigating further is the use of these tools together to learn environment features that assist robots with dynamic path planning. This approach

could help us weigh the pros and cons of creating a new roadmap whenever there is minimal change. Additionally, we are interested in advanced automation heuristics development that can simultaneously learn semantic features of the environment through topology roadmap and predict waypoints using active learning for feasible path planning.

In addition, we also plan to extend our topology-aware framework in various other ways. Firstly, we believe that our framework extension to the Task and Motion Planning (TAMP) problem can be fruitful, where the articulated linkage robot's arms control mechanism can be studied using topology features to determine infeasibility or failure in the symbolic search graphs at the task planning level. Secondly, the approach can extend to multi-agent cooperation for disaster-relief scenarios and warehouse settings. Our framework could provide a general path for each agent to follow in the planning space and subsequently utilize this information to communicate within the agent group to derive precise motions.

Glossary

$\mathcal{C}_{free}$ A set of all collision-free configurations. 23

$\mathcal{C}_{space}$ A space consisting of all possible robot configurations. 8, 14, 23, 36

conformation The spatial arrangements that the atoms in a molecule may adopt and freely convert between, especially by rotation about individual single bonds. 94, 112

convex hull The smallest convex polygon that encloses all of the configuration points in the set. 28, 39, 64, 68

critical points A set of non-degenerate points that define the convexity and concavity of a polygon. 55, 58, 67, 95, 105, 118

discrete Morse theory The discrete analog of classical smooth Morse theory. 118

Hausdorff distance The measure of distance between two subsets of a metric space from each other. 28, 69, 90, 106

homotopy classes It is a way of classifying geometric regions by studying the different types of paths that can be drawn in the region. 67

simplex The notion of the simplest possible polytope in any given dimension. 28

simplicial collapse It is an operation that shrinks simplicial complexes to homotopy-equivalent sub-complexes. 28, 34, 96, 105

simplicial complex It is a generalization of a graph representing higher-than-pairwise connectivity relationships. 55, 95, 105

Vietoris-Rips complex It is a way of forming a simplicial complex from distances using a set of points. 13, 23, 36, 55, 68, 95, 118

Acronyms

CBP CREB-binding Protein. 114

CREB cAMP-response element binding. 121

DMT discrete Morse theory. 36

IDPs Intrinsically Disordered Proteins. 105, 109, 110, 114

MKK4 dual specificity Mitogen-activated protein Kinase Kinase 4. 114

MPV Minimum Path Violation. xiii, xvi, 80, 84, 86, 88, 90, 118

PRM Probabilistic RoadMap. 8, 28, 35, 40, 43, 61, 101

RRT Rapidly exploring Random Trees. 9, 43

SBMP Sampling-Based Motion Planner. 45

TDA Topological Data Analysis. 4, 5, 7

BIBLIOGRAPHY

[1] Hamid Abdi and Saeid Nahavandi. Well-conditioned configurations of fault-tolerant manipulators. *Robotics and autonomous systems*, 60(2):242–251, 2012.

[2] Ercan U Acar and Howie Choset. Sensor-based coverage of unknown environments: Incremental construction of morse decompositions. *The International Journal of Robotics Research*, 21(4):345–366, 2002.

[3] Ercan U Acar, Howie Choset, Alfred A Rizzi, Prasad N Atkar, and Douglas Hull. Morse decompositions for coverage tasks. *The international journal of robotics research*, 21(4):331–344, 2002.

[4] Michal Adamaszek, Henry Adams, Ellen Gasparovic, Maria Gommel, Emilie Purvine, Radmila Sazdanovic, Bei Wang, Yusu Wang, and Lori Ziegelmeier. Vietoris-Rips and Čech complexes of metric gluings. In *34th International Symposium on Computational Geometry (SoCG 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[5] Torin Adamson, Julian Antolin Camarena, Lydia Tapia, and Bruna Jacobson. Optimizing low energy pathways in receptor-ligand binding with motion planning. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2041–2048. IEEE, 2019.

[6] Ibrahim Al-Bluwi, Thierry Siméon, and Juan Cortés. Motion planning algorithms for molecular simulations: A survey. *Computer Science Review*, 6(4):125–143, 2012.

[7] Jane R Allison, Peter Varnai, Christopher M Dobson, and Michele Vendruscolo. Determination of the free energy landscape of α-synuclein using spin label nuclear magnetic resonance measurements. *Journal of the American Chemical Society*, 131(51):18314–18326, 2009.

[8] Randa Almadhoun, Tarek Taha, Dongming Gan, Jorge Dias, Yahya Zweiri, and Lakmal Seneviratne. Coverage path planning with adaptive viewpoint sampling to construct 3d models of complex structures for the purpose of inspection. In *2018*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7047–7054. IEEE, 2018.

[9] Ahmad A Almarkhi, Anthony A Maciejewski, and Edwin KP Chong. An algorithm to design redundant manipulators of optimally fault-tolerant kinematic structure. *IEEE Robotics and Automation Letters*, 5(3):4727–4734, 2020.

[10] Dinler A Antunes, Jayvee R Abella, Didier Devaurs, Maurício M Rigo, and Lydia E Kavraki. Structure-based methods for binding mode and binding affinity prediction for peptide-mhc complexes. *Current topics in medicinal chemistry*, 18(26):2239–2255, 2018.

[11] Munehito Arai, Kenji Sugase, H Jane Dyson, and Peter E Wright. Conformational propensities of intrinsically disordered proteins influence the mechanism of binding and folding. *Proceedings of the National Academy of Sciences*, 112(31):9614–9619, 2015.

[12] Prasad N Atkar, Howie Choset, Alfred A Rizzi, and Ercan U Acar. Exact cellular decomposition of closed orientable surfaces embedded in $\mathfrak{R}^3$. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 1, pages 699–704. IEEE, 2001.

[13] Dominique Attali, André Lieutier, and David Salinas. Vietoris–Rips complexes also provide topologically correct reconstructions of sampled shapes. *Computational Geometry*, 46(4):448–465, 2013.

[14] M Madan Babu, Robin van der Lee, Natalia Sanchez de Groot, and Jörg Gsponer. Intrinsically disordered proteins: regulation and disease. *Current opinion in structural biology*, 21(3):432–440, 2011.

[15] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021.

[16] Mohamed Baioumy, Corrado Pezzato, Carlos Hernández Corbato, Nick Hawes, and Riccardo Ferrari. Towards stochastic fault-tolerant control using precision learning

and active inference. In *Machine Learning and Principles and Practice of Knowledge Discovery in Databases: International Workshops of ECML PKDD 2021, Virtual Event, September 13-17, 2021, Proceedings, Part I*, pages 681–691. Springer, 2022.

[17] Peter W Bates, Guo-Wei Wei, and Shan Zhao. Minimal molecular surfaces and their applications. *Journal of Computational Chemistry*, 29(3):380–391, 2008.

[18] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The Protein Data Bank. *Nucleic acids research*, 28(1):235–242, 2000.

[19] Frances C Bernstein, Thomas F Koetzle, Graheme JB Williams, Edgar F Meyer Jr, Michael D Brice, John R Rodgers, Olga Kennard, Takehiko Shimanouchi, and Mitsuo Tasumi. The Protein Data Bank: a computer-based archival file for macromolecular structures. *Journal of molecular biology*, 112(3):535–542, 1977.

[20] Subhrajit Bhattacharya. Search-based path planning with homotopy class constraints. In *Proceedings of the AAAI conference on artificial intelligence*, volume 24, pages 1230–1237, 2010.

[21] Subhrajit Bhattacharya. Identification and representation of homotopy classes of trajectories for search-based path planning in 3d. *Robotics: Science and Systems*, 2011.

[22] Subhrajit Bhattacharya, Robert Ghrist, and Vijay Kumar. Persistent homology for path planning in uncertain environments. *IEEE Transactions on Robotics*, 31(3):578–590, 2015.

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

[24] Subhrajit Bhattacharya, David Lipsky, Robert Ghrist, and Vijay Kumar. 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, 2013.

[25] Anders Björner. Topological methods. *Handbook of combinatorics*, 2:1819–1872, 1995.

[26] Robert Bogue. Disaster relief, and search and rescue robots: the way forward. *Industrial Robot: the international journal of robotics research and application*, 46(2):181–187, 2019.

[27] Jean-Daniel Boissonnat, Siddharth Pritam, and Divyansh Pareek. Strong collapse and persistent homology. *Journal of Topology and Analysis*, 15(01):185–213, 2023.

[28] V. Boor, M. H. Overmars, and A. F. van der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1018–1023, May 1999.

[29] Martim Brandão, Gerard Canal, Senka Krivić, and Daniele Magazzeni. Towards providing explanations for robot motion planning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3927–3933, 2021.

[30] Michael S Branicky, Ross A Knepper, and James J Kuffner. Path and trajectory diversity: Theory and algorithms. In *2008 IEEE International Conference on Robotics and Automation*, pages 1359–1364. IEEE, 2008.

[31] Glen E Bredon. *Sheaf theory*, volume 170. Springer Science & Business Media, 2012.

[32] Patrick Bryant, Gabriele Pozzati, and Arne Elofsson. Improved prediction of protein-protein interactions using alphafold2. *Nature communications*, 13(1):1–11, 2022.

[33] Tauã M Cabreira, Lisane B Brisolara, and Paulo R Ferreira Jr. Survey on coverage path planning with unmanned aerial vehicles. *Drones*, 3(1):4, 2019.

[34] Zixuan Cang, Lin Mu, and Guo-Wei Wei. Representability of algebraic topology for biomolecules in machine learning based scoring and virtual screening. *PLoS computational biology*, 14(1):e1005929, 2018.

[35] Arturo Casadevall and Liise-anne Pirofski. Host-pathogen interactions: basic concepts of microbial commensalism, colonization, infection, and disease. *Infection and immunity*, 68(12):6511–6518, 2000.

[36] Erin W Chambers, Vin De Silva, Jeff Erickson, and Robert Ghrist. Vietoris–Rips complexes of planar point sets. *Discrete & Computational Geometry*, 44(1):75–90, 2010.

[37] Cyril Charlier, Guillaume Bouvignies, Philippe Pelupessy, Astrid Walrant, Rodrigue Marquant, Mikhail Kozlov, Pablo De Ioannes, Nicolas Bolik-Coulon, Sandrine Sagan, Patricia Cortes, et al. Structure and dynamics of an intrinsically disordered protein region that partially folds upon binding by chemical-exchange nmr. *Journal of the American Chemical Society*, 139(35):12219–12227, 2017.

[38] Theodoros Chondrogiannis, Panagiotis Bouros, Johann Gamper, Ulf Leser, and David B Blumenthal. Finding k-dissimilar paths with minimum collective length. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 404–407, 2018.

[39] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A. Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, June 2005.

[40] Veronika Csizmok, Ariele Viacava Follis, Richard W Kriwacki, and Julie D Forman-Kay. Dynamic protein interaction networks and new structural paradigms in signaling. *Chemical reviews*, 116(11):6424–6462, 2016.

[41] Sebastian Daberdaku and Carlo Ferrari. Antibody interface prediction with 3d Zernike descriptors and SVM. *Bioinformatics*, 35(11):1870–1876, 2019.

[42] Sjoerd J de Vries, Christina EM Schindler, Isaure Chauvot de Beauchêne, and Martin Zacharias. A web interface for easy flexible protein-protein docking with attract. *Biophysical journal*, 108(3):462–465, 2015.

[43] Elise Delaforge, Jaka Kragelj, Laura Tengo, Andrés Palencia, Sigrid Milles, Guillaume Bouvignies, Nicola Salvi, Martin Blackledge, and Malene Ringkjøbing Jensen. Deciphering the dynamic interaction profile of an intrinsically disordered protein by nmr exchange spectroscopy. *Journal of the American Chemical Society*, 140(3):1148–1158, 2018.

[44] Jory Denny, Read Sandström, Andrew Bregger, and Nancy M Amato. Dynamic region-biased rapidly-exploring random trees. In *Twelfth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.

[45] Israel T Desta, Kathryn A Porter, Bing Xia, Dima Kozakov, and Sandor Vajda. Performance and its limits in rigid body protein-protein docking. *Structure*, 28(9):1071–1081, 2020.

[46] Didier Devaurs, Dinler A Antunes, Sarah Hall-Swan, Nicole Mitchell, Mark Moll, Gregory Lizée, and Lydia E Kavraki. Using parallelized incremental meta-docking can solve the conformational sampling issue when docking large ligands to proteins. *BMC molecular and cell biology*, 20(1):1–15, 2019.

[47] Tamal Krishna Dey, Fengtao Fan, and Yusu Wang. Graph induced complex on point data. In *Proceedings of the twenty-ninth annual symposium on Computational geometry*, pages 107–116, 2013.

[48] Andrew Dobson and Kostas E Bekris. Improving sparse roadmap spanners. In *2013 IEEE International Conference on Robotics and Automation*, pages 4106–4111. IEEE, 2013.

[49] Herbert Edelsbrunner and Dmitriy Morozov. Persistent homology: theory and practice. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2012.

[50] David Eppstein, Maarten Löffler, and Darren Strash. Listing all maximal cliques in large sparse real-world graphs. *Journal of Experimental Algorithmics (JEA)*, 18:3–1, 2013.

[51] David Eppstein and Darren Strash. Listing all maximal cliques in large sparse real-world graphs. In *International Symposium on Experimental Algorithms*, pages 364–375. Springer, 2011.

[52] Lawrence H Erickson and Steven M LaValle. Survivability: Measuring and ensuring path diversity. In *2009 IEEE International Conference on Robotics and Automation*, pages 2068–2073. IEEE, 2009.

[53] Alejandro Estana. *Algorithms and comptutational tools for the study of Intrinsically Disordered Proteins*. PhD thesis, Toulouse, INSA, 2020.

[54] Xin Feng, Kelin Xia, Zhan Chen, Yiying Tong, and Guo-Wei Wei. Multiscale geometric modeling of macromolecules ii: Lagrangian representation. *Journal of computational chemistry*, 34(24):2100–2120, 2013.

[55] Leonardo Fermin-Leon, José Neira, and José A Castellanos. Tigre: Topological graph based robotic exploration. In *2017 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2017.

[56] Robin Forman. A user's guide to discrete Morse theory. *Sém. Lothar. Combin*, 48:35pp, 2002.

[57] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.

[58] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Informed RRT: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2997–3004. IEEE, 2014.

[59] Michael K Gilson and Huan-Xiang Zhou. Calculation of protein-ligand binding affinities. *Annu. Rev. Biophys. Biomol. Struct.*, 36:21–42, 2007.

[60] S. Gottschalk, M.C. Lin, and D. Manocha. Obb-tree: A hierarchical structure for rapid interference detection. Technical Report TR96-013, University of N. Carolina, Chapel Hill, CA, 1996.

[61] K. Hauser. Lazy collision checking in asymptotically-optimal motion planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2951–2957, May 2015.

[62] Kris Hauser. The minimum constraint removal problem with three robotics applications. *The International Journal of Robotics Research*, 33(1):5–17, 2014.

[63] Penny A Holding and Robert W Snow. Impact of plasmodium falciparum malaria on performance and learning: review of the evidence. *The Intolerable Burden of Malaria: A New Look at the Numbers: Supplement to Volume 64 (1) of the American Journal of Tropical Medicine and Hygiene*, 2001.

[64] D. Hsu, T. Jiang, J.H. Reif, and Z. Sun. Bridge test for sampling narrow passages with probabilistic roadmap planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4420–4426. IEEE, 2003.

[65] Malene Ringkjøbing Jensen, Rob WH Ruigrok, and Martin Blackledge. Describing intrinsically disordered proteins at atomic resolution by nmr. *Current opinion in structural biology*, 23(3):426–435, 2013.

[66] S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[67] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.

[68] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[69] Sebastian Klemm, Jan Oberländer, Andreas Hermann, Arne Roennau, Thomas Schamm, J Marius Zollner, and Rüdiger Dillmann. RRT*-connect: Faster, asymptotically optimal motion planning. In *2015 IEEE international conference on robotics and biomimetics (ROBIO)*, pages 1670–1677. IEEE, 2015.

[70] Ross A Knepper and Matthew T Mason. Path diversity is only part of the problem. In *2009 IEEE International Conference on Robotics and Automation*, pages 3224–3229. IEEE, 2009.

[71] Vinay Krishnaswamy and Wyatt S Newman. Online motion planning using critical point graphs in two-dimensional configuration space. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2334–2339. IEEE, 1992.

[72] KUKA Robotics Corporation. Kukayoubot. https://cyberbotics.com/doc/guide/youbot. Accessed: January 7, 2022.

[73] Prakash Kulkarni and Vladimir N Uversky. Intrinsically disordered proteins in chronic diseases, 2019.

[74] Parasol Lab. Parasol planning library. `https://github.com/parasol-ppl/ppl`, 2022.

[75] Fabien Lagriffoul and Benjamin Andres. Combining task and motion planning: A culprit detection problem. *The International Journal of Robotics Research*, 35(8):890–927, 2016.

[76] Przemyslaw A Lasota and Julie A Shah. Analyzing the effects of human-aware motion planning on close-proximity human–robot collaboration. *Human factors*, 57(1):21–33, 2015.

[77] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–479, 1999.

[78] Steven M LaValle et al. Rapidly-exploring random trees: A new tool for path planning, 1998.

[79] John Lhota and Lei Xie. Protein-fold recognition using an improved single-source k diverse shortest paths algorithm. *Proteins: Structure, Function, and Bioinformatics*, 84(4):467–472, 2016.

[80] Sihui Li and Neil T. Dantam. Towards general infeasibility proofs in motion planning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6704–6710, 2020.

[81] Zhan Li, Chunxu Li, Shuai Li, and Xinwei Cao. A fault-tolerant method for motion planning of industrial redundant manipulator. *IEEE transactions on industrial informatics*, 16(12):7469–7478, 2019.

[82] Tomas Lozano-Perez. Spatial planning: A configuration space approach. *Computers, IEEE Transactions on*, 100(2):108–120, 1983.

[83] Haiying Lu, Qiaodan Zhou, Jun He, Zhongliang Jiang, Cheng Peng, Rongsheng Tong, and Jianyou Shi. Recent advances in the development of protein–protein interactions modulators: mechanisms and clinical trials. *Signal transduction and targeted therapy*, 5(1):1–23, 2020.

[84] Anthony A Maciejewski. Fault tolerant properties of kinematically redundant manipulators. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 638–642. IEEE, 1990.

[85] Chiara Maniaci and Alessio Ciulli. Bifunctional chemical probes inducing protein–protein interactions. *Current Opinion in Chemical Biology*, 52:145–156, 2019.

[86] Reza Mashayekhi, Mohd Yamani Idna Idris, Mohammad Hossein Anisi, Ismail Ahmedy, and Ihsan Ali. Informed RRT*-connect: An asymptotically optimal single-query path planning method. *IEEE Access*, 8:19842–19852, 2020.

[87] Zoe McCarthy, Timothy Bretl, and Seth Hutchinson. Proving path non-existence using sampling and alpha shapes. In *2012 IEEE International Conference on Robotics and Automation*, pages 2563–2569, 2012.

[88] Troy McMahon, Sam Jacobs, Bryan Boyd, Lydia Tapia, and Nancy M. Amato. Evaluation of the k-closest neighbor selection strategy for PRM construction. Technical Report TR12-002, Texas A&M, College Station Tx., 2011.

[89] Xiangmao Meng, Ju Xiang, Ruiqing Zheng, Fang-Xiang Wu, and Min Li. Dpcmne: detecting protein complexes from protein-protein interaction networks via multi-level network embedding. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(3):1592–1602, 2021.

[90] Sigrid Milles, Nicola Salvi, Martin Blackledge, and Malene Ringkjøbing Jensen. Characterization of intrinsically disordered proteins and their dynamic complexes: From in vitro to cell-like environments. *Progress in nuclear magnetic resonance spectroscopy*, 109:79–100, 2018.

[91] Elizabeth Munch. A user's guide to topological data analysis. *Journal of Learning Analytics*, 4(2):47–61, 2017.

[92] Timo Oksanen and Arto Visala. Coverage path planning algorithms for agricultural field machines. *Journal of field robotics*, 26(8):651–668, 2009.

[93] Andreas Orthey, Sohaib Akbar, and Marc Toussaint. Multilevel motion planning: A fiber bundle formulation. *The international journal of robotics research*, 2020.

[94] Andreas Orthey, Benjamin Frész, and Marc Toussaint. Motion planning explorer: Visualizing local minima using a local-minima tree. *IEEE Robotics and Automation Letters*, 5(2):346–353, 2019.

[95] Luigi Palmieri, Andrey Rudenko, and Kai O Arras. A fast random walk approach to find diverse paths for robot navigation. *IEEE Robotics and Automation Letters*, 2(1):269–276, 2016.

[96] Kris Pauwels, Pierre Lebrun, and Peter Tompa. To be disordered or not to be disordered: is that still a question for proteins in the cell? *Cellular and Molecular Life Sciences*, 74(17):3185–3204, 2017.

[97] Eric F Pettersen, Thomas D Goddard, Conrad C Huang, Gregory S Couch, Daniel M Greenblatt, Elaine C Meng, and Thomas E Ferrin. UCSF Chimera—a visualization system for exploratory research and analysis. *Journal of computational chemistry*, 25(13):1605–1612, 2004.

[98] Farzin Piltan, Alexander E Prosvirin, Muhammad Sohaib, Belem Saldivar, and Jong-Myon Kim. An SVM-based neural adaptive variable structure observer for fault diagnosis and fault-tolerant control of a robot manipulator. *Applied Sciences*, 10(4):1344, 2020.

[99] Erion Plaku and Lydia E Kavraki. Distributed sampling-based roadmap of trees for large-scale motion planning. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3868–3873. IEEE, 2005.

[100] Florian T Pokorny, Majd Hawasly, and Subramanian Ramamoorthy. Multiscale topological trajectory classification with persistent homology. In *Robotics: science and systems*, 2014.

[101] Florian T Pokorny and Danica Kragic. Data-driven topological motion planning with persistent cohomology. In *Robotics: Science and Systems*, 2015.

[102] Michael Robinson. Understanding networks and their behaviors using sheaf theory. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, pages 911–914. IEEE, 2013.

[103] Julian Ryde, Vikas Dhiman, and Robert Platt. Voxel planes: Rapid visualization and meshification of point cloud ensembles. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3731–3737. IEEE, 2013.

[104] Rushikesh Sable and Seetharama Jois. Surfing the protein-protein interaction surface using docking methods: application to the design of ppi inhibitors. *Molecules*, 20(6):11569–11603, 2015.

[105] Read Sandström, Diane Uwacu, Jory Denny, and Nancy M Amato. Topology-guided roadmap construction with dynamic region sampling. *IEEE Robotics and Automation Letters*, 5(4):6161–6168, 2020.

[106] Robert Schneider, Martin Blackledge, and Malene Ringkjøbing Jensen. Elucidating binding mechanisms and dynamics of intrinsically disordered protein complexes using nmr spectroscopy. *Current opinion in structural biology*, 54:10–18, 2019.

[107] Donald R Sheehy. Linear-size approximations to the Vietoris–Rips filtration. *Discrete & Computational Geometry*, 49(4):778–796, 2013.

[108] Vahid Sheikhhassani, Barbara Scalvini, Julian Ng, Laurens WHJ Heling, Yosri Ayache, Tom MJ Evers, Eva Estébanez-Perpiñá, Iain J McEwan, and Alireza Mashaghi. Topological dynamics of an intrinsically disordered n-terminal domain of the human androgen receptor. *Protein Science*, 31(6):e4334, 2022.

[109] Kensen Shi, Jory Denny, and Nancy M Amato. Spark PRM: Using RRTs within PRMs to efficiently explore narrow passages. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4659–4666. IEEE, 2014.

[110] Nir Shvalb, Boaz Ben Moshe, and Oded Medina. A real-time motion planning algorithm for a hyper-redundant set of mechanisms. *Robotica*, 31(8):1327–1335, 2013.

[111] Graham R Smith and Michael JE Sternberg. Prediction of protein-protein interactions by docking methods. *Current opinion in structural biology*, 12(1):28–35, 2002.

[112] Sharon Sunny and PB Jayaraj. Protein–protein docking: Past, present, and future. *The protein journal*, pages 1–26, 2022.

[113] Markku Suomalainen, Yiannis Karayiannidis, and Ville Kyrki. A survey of robot manipulation in contact. *Robotics and Autonomous Systems*, 156:104224, 2022.

[114] Autumn R Tobin, Rachel Crow, Darya V Urusova, Jason C Klima, Niraj H Tolia, and Eva-Maria Strauch. Inhibition of a malaria host–pathogen interaction by a computationally designed inhibitor. *Protein Science*, 32(1):e4507, 2023.

[115] Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, 363(1):28–42, 2006.

[116] Peter Tompa, Eva Schad, Agnes Tantos, and Lajos Kalmar. Intrinsically disordered proteins: emerging interaction specialists. *Current opinion in structural biology*, 35:49–59, 2015.

[117] Maxim Totrov and Ruben Abagyan. Flexible ligand docking to multiple receptor conformations: a practical alternative. *Current opinion in structural biology*, 18(2):178–184, 2008.

[118] Aakriti Upadhyay and Chinwe Ekenna. Investigating heterogeneous planning spaces. In *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, pages 108–115. IEEE, 2018.

[119] Aakriti Upadhyay and Chinwe Ekenna. A geometric and topological analysis of the binding behavior of intrinsically disordered proteins. In *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 3504–3511. IEEE, 2022.

[120] Aakriti Upadhyay and Chinwe Ekenna. A new tool to study the binding behavior of intrinsically disordered proteins. *International Journal of Molecular Sciences*, 24(14), 2023.

[121] Aakriti Upadhyay, Mukulika Ghosh, and Chinwe Ekenna. Minimal Path Violation problem with application to fault tolerant motion planning of manipulators. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023.

[122] Aakriti Upadhyay, Boris Goldfarb, and Chinwe Ekenna. A topological approach to finding coarsely diverse paths. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6552–6557. IEEE, 2021.

[123] Aakriti Upadhyay, Boris Goldfarb, and Chinwe Ekenna. Incremental path planning algorithm via topological mapping with metric gluing. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1290–1296. IEEE, 2022.

[124] Aakriti Upadhyay, Boris Goldfarb, Weifu Wang, and Chinwe Ekenna. A new application of discrete Morse theory to optimizing safe motion planning paths. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 18–35. Springer, 2022.

[125] Aakriti Upadhyay, Tuan Tran, and Chinwe Ekenna. A topology approach towards modeling activities and properties on a biomolecular surface. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 157–162. IEEE, 2021.

[126] Aakriti Upadhyay, Weifu Wang, and Chinwe Ekenna. Approximating c-free space topology by constructing Vietoris-Rips complex. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2517–2523. IEEE, 2019.

[127] Vladimir N Uversky, Christopher J Oldfield, and A Keith Dunker. Intrinsically disordered proteins in human diseases: introducing the D^2 concept. *Annual review of biophysics*, 37(1):215–246, 2008.

[128] Ilya A Vakser. Protein-protein docking: From interaction to interactome. *Biophysical journal*, 107(8):1785–1793, 2014.

[129] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 50(D1):D439–D444, 2022.

[130] Anastasiia Varava, J. Frederico Carvalho, Florian T. Pokorny, and Danica Kragic. Caging and path non-existence: A deterministic sampling-based verification algorithm.

In Nancy M. Amato, Greg Hager, Shawna Thomas, and Miguel Torres-Torriti, editors, *Robotics Research*, pages 589–604, Cham, 2020. Springer International Publishing.

[131] Arun T Vemuri, Marios M Polycarpou, and Sotiris A Diakourtis. Neural network based fault detection in robotic manipulators. *IEEE transactions on Robotics and Automation*, 14(2):342–348, 1998.

[132] Hiparco Lins Vieira, Eric Wajnberg, André Teófilo Beck, and Maira Martins da Silva. Reliable motion planning for parallel manipulators. *Mechanism and Machine Theory*, 140:553–566, 2019.

[133] Vojtěch Vonásek, Adam Jurčík, Katarína Furmanová, and Barbora Kozlíková. Sampling-based motion planning for tracking evolution of dynamic tunnels in molecular dynamics simulations. *Journal of Intelligent & Robotic Systems*, 93:763–785, 2019.

[134] Vojtěch Vonásek, Robert Pěnička, and Barbora Kozlíková. Computing multiple guiding paths for sampling-based motion planning. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 374–381. IEEE, 2019.

[135] Caleb Voss, Mark Moll, and Lydia E Kavraki. A heuristic approach to finding diverse short paths. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4173–4179. IEEE, 2015.

[136] Gaoqi Weng, Ercheng Wang, Zhe Wang, Hui Liu, Feng Zhu, Dan Li, and Tingjun Hou. HawkDock: a web server to predict and analyze the protein–protein complex based on computational docking and MM/GBSA. *Nucleic acids research*, 47(W1):W322–W330, 2019.

[137] Wikipedia. Combination. `https://en.wikipedia.org/wiki/Combination`, 2023. Accessed: 2023-02-18.

[138] Wikipedia. Hausdorff distance, 2023. Accessed: 2023-02-21.

[139] Wikipedia. Permutation. `https://en.wikipedia.org/wiki/Permutation`, 2023. Accessed: 2023-02-18.

[140] WillowGarage. PR2 Robot. www.willowgarage.com, 2016.

[141] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1024–1031, 1999.

[142] Peter E Wright and H Jane Dyson. Intrinsically disordered proteins in cellular signalling and regulation. *Nature reviews Molecular cell biology*, 16(1):18–29, 2015.

[143] Y. Wu. An obstacle-based probabilistic roadmap method for path planning. Master's thesis, Department of Computer Science, Texas A&M University, 1996.

[144] Kelin Xia, Xin Feng, Zhan Chen, Yiying Tong, and Guo-Wei Wei. Multiscale geometric modeling of macromolecules i: Cartesian representation. *Journal of Computational Physics*, 257:912–936, 2014.

[145] Yumeng Yan, Huanyu Tao, Jiahua He, and Sheng-You Huang. The HDOCK server for integrated protein–protein docking. *Nature protocols*, 15(5):1829–1852, 2020.

[146] Zichen Yan, Junbo Tan, Bin Liang, Houde Liu, and Jun Yang. Active fault-tolerant control integrated with reinforcement learning application to robotic manipulator. In *2022 American Control Conference (ACC)*, pages 2656–2662. IEEE, 2022.

[147] Anna Yershova, Leonard Jaillet, Thierry Simeon, and Steven M. Lavalle. Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3856–3861, April 2005.

[148] Daqing Yi, Michael A Goodrich, Thomas M Howard, and Kevin D Seppi. Topology-aware RRT* for parallel optimal sampling in topologies. In *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on*, pages 513–518. IEEE, 2017.

[149] Shuangye Yin, Elizabeth A Proctor, Alexey A Lugovskoy, and Nikolay V Dokholyan. Fast screening of protein surfaces using geometric invariant fingerprints. *Proceedings of the National Academy of Sciences*, 106(39):16622–16626, 2009.

[150] Qiangfeng Cliff Zhang, Donald Petrey, Lei Deng, Li Qiang, Yu Shi, Chan Aye Thu, Brygida Bisikirska, Celine Lefebvre, Domenico Accili, Tony Hunter, et al. Structure-based prediction of protein–protein interactions on a genome-wide scale. *Nature*, 490(7421):556–560, 2012.

[151] Xiaolei Zhu, Yi Xiong, and Daisuke Kihara. Large-scale binding ligand prediction by improved patch-based method Patch-Surfer2. 0. *Bioinformatics*, 31(5):707–713, 2015.

[152] Afra Zomorodian. Fast construction of the Vietoris–Rips complex. *Computers & Graphics*, 34(3):263–271, 2010.

[153] Afra Zomorodian. Topological data analysis. *Advances in applied and computational topology*, 70:1–39, 2012.

[154] Meghan Zuck, Laura S Austin, Samuel A Danziger, John D Aitchison, and Alexis Kaushansky. The promise of systems biology approaches for revealing host pathogen interactions in malaria. *Frontiers in microbiology*, 8:2183, 2017.