J Intell Robot Syst (2016) 82:555–575
DOI 10.1007/s10846-015-0278-z

Homotopic Roadmap Generation for Robot Motion Planning

Rahul Kala

Received: 1 April 2015 / Accepted: 15 September 2015 / Published online: 1 October 2015

Abstract Two robot paths are said to be in the same homotopic group if one can be obtained from the other by multiple small deformations. Knowledge of robot homotopic groups gives information regarding the obstacle structure and enables timely computation of optimal path. Making a roadmap which misses out on a single homotopic group in such approaches may lead to sub-optimal decisions. E.g. one may prefer to go through a very narrow corridor if that reduces the path length significantly, but not if the resulting path has too many such narrow segments. Similarly knowledge of homotopic groups may enable distribution and scheduling of robots across homotopic groups for decentralized planning of multiple robots. For an unstructured robot environment, sampling based approaches give an insight into homotopic groups. The aim of the work is to make a homotopy conscious Probabilistic Roadmap such that the roadmap is capable of generating paths corresponding to as many homotopic groups as possible. Experimental results confirm that the proposed approach gives the best results as compared to the other sampling techniques subject to the test scenarios run.

Keywords Sampling · Robot motion planning · Homotopy identification · Probabilistic roadmap · Multi-robotics

Mathematics Subject Classification (2010) 68T40 Robotics

R. Kala (✉)
Robotics and Artificial Intelligence Laboratory, Indian Institute of Information Technology Allahabad, Allahabad, India
e-mail: rkala001@gmail.com; http://rkala.in/

1 Introduction

Mobile robots are expected to navigate from one state to the other in order to perform various useful tasks usually in home or office environments. Let the configuration space of the robot be given by C. A robotic environment may be filled with multiple obstacles which need to be avoided while carrying out any navigation. Robot states such that any part of the robot is in collision with any obstacle are given by the obstacle prone configuration space denoted by C_{obs}. The obstacle-free configuration space, $C_{free} = C \backslash C_{obs}$, hence denotes the states that a robot may take at any time during its operation. The problem of path planning of a robot deals with finding a path τ from a source to a goal. The source state ($s_r \in C_{free}$) and the goal state ($g_r \in C_{free}$) is known in advance. The path (τ) denotes the continuum of collision-free states

from source to goal state, given by, $\tau : [0, 1] \rightarrow C_{\text{free}}$, $\tau(0) = s_r$, $\tau(1) = g_r$. Here $\tau(s)$ denotes any general state in the robot path.

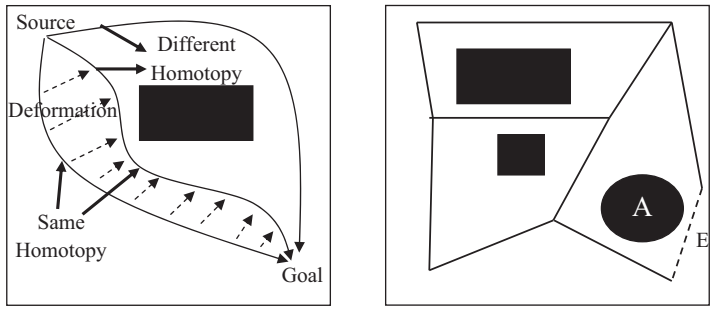
Infinitely many collision-free paths may be possible from the source to the goal. Each path is a specification of the manner of avoiding obstacles at a coarser level and specific distances to maintain from the obstacles at a finer level. The obstacles divide the set of paths into groups, each group or homotopy denoting a specific strategy of overcoming obstacles. The concept is shown in Fig. 1a. Two paths and are said to be in the same homotopic group if one can be deformed to the other by multiple infinitely small deformations, such that all intermediate paths are collision-free. Let τ_1: [0,1]$\rightarrow$ C_{free} and τ_2: [0,1]$\rightarrow$ C_{free} be two paths with $\tau_1(0) = \tau_2(0) = s_r$ and $\tau_1(1) = \tau_2(1) = g_r$. A deformation/mapping function is defined as h: $h_0 = \tau_1$, $h_1 = \tau_2$, $h_t(0) = s_r$, $h_t(1) = g_r$. Here h_t represents the path at any general step t of deformation, $t \in$[0,1]. The paths are said to be in the same homotopic group if such a continuous mapping h exists from the first path $h_0 = \tau_1$ to the second path $h_1 = \tau_2$ such that all intermediate paths h_t are collision-free from source $h_t(0) = s_r$ to goal $h_t(1) = g_r$. A related term is homology. Two paths are said to be homologous to each other if the area bounded by the two paths (that is the area made by the first path and the reverse of the second path) in a 2 D manifold embedded in the configuration space is obstacle-free.

Here environment is assumed to be unstructured. By unstructured environment we mean that no structural information of the obstacles is available. Even though the complete environment is available, it is available in the form of high-resolution grid points specifying obstacle-prone and obstacle-free areas. Even though it is possible to use algorithms to extract structure out of an unstructured environment representation, such algorithms are themselves time consuming and prone to resolution loss which for the specific problem can be fatal.

Knowledge of homotopic groups in an unstructured irregular obstacle prone environment can lead to establishing a structure in C_{free}, which helps in complex decision making of the robotic mission. In generation of a roadmap it is important to consider whether the roadmap is able to sight nearly all homotopic groups for the given scenario of operation. This ensures that nearly all possible routes are discovered for optimal decision making. Consider the case of Fig. 1b, wherein any search algorithm cannot discover homotopic groups that go from the right of obstacle A. However once the edge E has been added to the roadmap, it is possible to sight all possible homotopic groups. A deformation retract is formed by shrinking all free spaces to a unit size and the resulting roadmap guarantees generation of all possible homotopic groups possible between any pair of source and goal. The deformation retract with edges equidistant to obstacles may not always be the best roadmap and may be very hard to generate in a high-resolution unstructured setting. It further does not allow redundant edges. Here a more generalized term of homotopy conscious roadmap is used to refer to roadmaps which are capable of generating as many homotopic groups as possible, subjected to the constraints of computation time.

Optimality of decision making may hence be largely attributed to whether all homotopic groups can be discovered. Missing out a single homotopic group can be harmful if the optimal plan involved the missed homotopic group while the missed homotopic group was difficult to find. For the task of path planning,

Fig. 1 Homotopic groups



(a) Trajectories in same and different homotopic groups **(b) Roadmap capable of generating all homotopic groups**

the path costs may nearly remain the same for all paths within a homotopic group, while the costs may significantly differ for different homotopic groups. This is with the assumption that there are no highly wide open spaces for the optimal path in which case the planning is relatively easy. The choice of homotopic group or path cost may be based on a variety of factors including the number of turns, smoothness, clearance, path length, operational speed constraints, terrain, uncertainty and dynamics of the environment, etc. In motion planning computation of the optimal homotopic group is more difficult in comparison to the computation of the optimal path within a pre-specified homotopic group. Consider the case wherein the optimal path consists of going through a narrow corridor as a short-cut to taking a long encircled route. The narrow corridors are hard to find for sampling based approaches. The algorithm is likely to miss out sighting the corridor and instead take the long route around the obstacles. However, the narrow corridor may be wrong to take if it involves a lot of tight maneuvers, thus risking collision unless the robot is very slow and cautious.

This work is inspired from the earlier works of the author in the problem of multi-robot path planning [1, 2], where the paths of all the robots need to be computed, such that no robot collides with all the others. A typical way of solving the problem is to choose the homotopic groups independently for all the robots considering only the static obstacles at a coarser level. The coarser level path is then used for finer level planning in a decentralised manner, wherein a coordination mechanism is applied for handling robot collisions. A major drawback of the approach is that the homotopic group selection is independent for all robots, and hence the coarser level planning can schedule a large number of robots in the same areas at the same times, leaving the other routes largely unused. All the robots on the popular routes hence have to make their way out in a crowded scenario which is sub-optimal.

A solution to the problem is to consider the other robots in the homotopic group assignment which may still be planned in a decentralized manner. Such an approach was used in [3] wherein homotopic group referred only to the static obstacles while the homotopic group assignments considered the motion of other robots; and approach used in [4] wherein homotopic group referred to both the static and dynamic obstacles (other robots). Both these solutions were however for road/traffic scenarios wherein there is a prior known structure in the form of a straight/curved road with all robots/vehicles travelling inbound or outbound. In the absence of this road structure, in a widely bounded robotic environment, the task of homotopic group computation is not naïve. Once the set of all homotopic groups are known, judicious assignment and scheduling of robots at various regions of the homotopic groups may be done.

The paper is aimed in making the roadmap generation conscious of homotopic groups for a given robotic environment. Probabilistic Roadmap (PRM) is used for getting a homotopic structure out of the environment, while a sampling technique is proposed that ensures that as many homotopic groups gets represented as possible in the constraints of time. Attempts are made to make the samples get generated at the best places, so as to have overall optimal plans of the robots. This corresponds to promoting obstacle-prone samples to the nearest obstacle-free areas, focussing on the obstacle boundary and maintaining a reasonable separation of the sample from obstacle boundary if possible. The aim is to discover as many homotopic groups as possible, especially the ones more difficult to discover, with the least number of roadmap vertices and edges.

The contributions of the paper are as follows: (i) The problem of discovering all homotopic groups for an unstructured environment is studied, which is a rather untouched problem in the literature with most solutions focussing on the structured counterpart. (ii) The concept of local roadmaps with size proportional to the difficulty of the region is devised. The local roadmap facilitates redundant connectivity to the rest of the roadmap. The approach can simultaneously cater to narrow corridors, congested environments and otherwise difficult areas. (iii) An edge connection strategy is formulated which discounts the connectivity of an edge based on the sizes of the roadmaps that it is directly or indirectly connected to. The strategy saves computational time at the same time maintaining nearly the same connectivity of the roadmap as a k-connected PRM. (iv) The design of the algorithm is made generic to suit a variety of cases including narrow corridors, open spaces, multiple pathways between obstacles of multiple widths and narrow corridors amidst wide open spaces. The aim is to have a common algorithm that performs well in all such mixed scenarios.

This paper is organized as follows. Section 2 presents the problem formulation, including presentation of a few basic concepts. Section 3 gives the related works. Section 4 presents the proposed algorithm. Results are given in Section 5. Section 6 gives some discussions. The conclusions are presented in Section 7.

2 Problem

2.1 Probabilistic Roadmap

PRM is a sampling based approach which relies upon generation of random samples in the configuration space, which become the vertices of the roadmap. Possibility of an edge between all pairs of vertices is checked by a local search technique. A typical way is to use straight line connections as the local search technique. This roadmap is computed offline for any robotic environment, which can then be used for online planning in the query phase. The number of edges per vertex may be restricted to reduce the computational time. The concentration is on edges of a vertex with the neighbouring vertices, the feasibility computation of which is relatively less expensive. A roadmap with lesser number of vertices and edges is more time-efficient in the online planning phase. The approach is probabilistically complete and probabilistically optimal. Practically the algorithm needs to be terminated after a reasonable time to limit the roadmap construction and online planning computation time.

One of the ways to restrict the number of edges is to check connectivity and add an edge from any vertex to a maximum of k closest neighbouring vertices [5], which is the method used in this paper. The selection of this method was done after due consideration to other popular methods which are very briefly critiqued here. A popular method is to check connectivity with any vertex within a radius of k [6]. The method works well for general PRM, however it is not used here since the number of vertices are very limited and some vertices (connecting two regions separated by a narrow corridor) may be reasonably far. Another method of restricting the edges is to use graph spanners or related methods [7, 8], which work after the roadmap construction, identifying and removing not very useful edges. A typical way is to remove edges if the shortest path between two vertices does not get worse by a factor of k times. As the aim of the work is to reduce the roadmap construction time at the first place, this method is not used. Iterative solutions to the same are possible [9], but the optimality of the roadmap in the initial few iterations cannot be ascertained. The same concept is extended to Adaptive Roadmaps [10, 11] and Elastic Roadmaps [12, 13] wherein the roadmaps can adapt to the changing environment.

A related method is Rapidly-exploring Random Trees (RRT) wherein the samples are not used directly as vertices of the roadmap, but are used to extend the roadmap towards the regions of the generated sample. The roadmap is in the form of a tree which is initiated with source as the only node. At every time a random sample is generated, the closest node in the tree is found, and the closest node is extended towards the sample by a modest step size; which forms a new node in the tree. For better performance, it is nice to use bidirectional search from both source and goal [14], extend a node until it reaches the end (RRT-Connect [15]), and use multiple trees [16–19].

2.2 Problem Formulation

This paper focuses on the problem of making roadmap generation conscious of the homotopic groups, such that nearly all homotopic groups can be represented. Since the problem is motivated from the problem of multi-robot motion planning, the definition of homotopy is extended to all sources and goals of all the robots. Let $G(V, E)$ be a roadmap graph where V denotes the vertices and E denotes the edges. Let $\{s_r \cup g_r\}$ be the set of sources and goals of all robots, which all need to be present in the graph ($\{s_r \cup g_r\} \subset V$). Note that the constraint of sources and goals pre-existing in the roadmap is only done to facilitate the definition of homotopy. Let $Path(V_1, V_2 \ldots ..V_n)$ denote the path from V_1 to V_n such that any two consecutive pairs of vertices V_i and V_{i+1} is an edge or $(V_i, V_{i+1}) \in E$. The aim is to have the least number of vertices ($|V|$) and preferably the least number of edges ($|E|$) in this graph such that all homotopic groups can be represented or Eq. 1.

$$\forall \tau, \exists Path(x_1 \ldots x_2) : Homotopic(\tau, Path(x_1 \ldots x_2)),$$
$$x_1, x_2 \in \{s_r \cup g_r\}, x_2 \neq x_1, \tau(0) = x_1, \tau(1) = x_2 \quad (1)$$

Here *Homotopic*(x, y) states if x and $y$are in the same homotopic group.

For a small number of samples or for difficult regions the approach may not be successful in making a roadmap which represents all homotopic groups. The metric designed to measure the performance of the approach is taken as the number of edges ($|\Gamma|$) to be added in the roadmap so that the resultant roadmap is capable of representing all possible homotopic groups. Here Γ denotes the collection of edges added. In simple terms, edges must exist all around all obstacles. If an edge is absent, the specific homotopic groups around that obstacle may not be generated. The roadmap can be completed by adding additional edges such that all homotopic groups are represented. The least number of such edges is the metric of study. Here an edge ($\tau \in \Gamma$) may denote a line, curve or any valid path whose both ends are vertices of the roadmap, and it does not touch or intersect any obstacle or any other edge. This is given by Eq. 2. Note that the number of homotopic groups is not recommendable as a metric since it is hard to understand and for any missing link such a metric can be infinite.

$$\tau : \tau(0), \tau(1) \in V, \tau(s) \in C_{free}, \tau(s) \neq e(t) \forall e \in E, \quad 0 \leq s, t \leq 1 \tag{2}$$

$$\min |\Gamma| \forall \tau \in \Gamma \text{ satisfies (2)}, \{E \cup \Gamma\} \text{satisfies (1)} \tag{3}$$

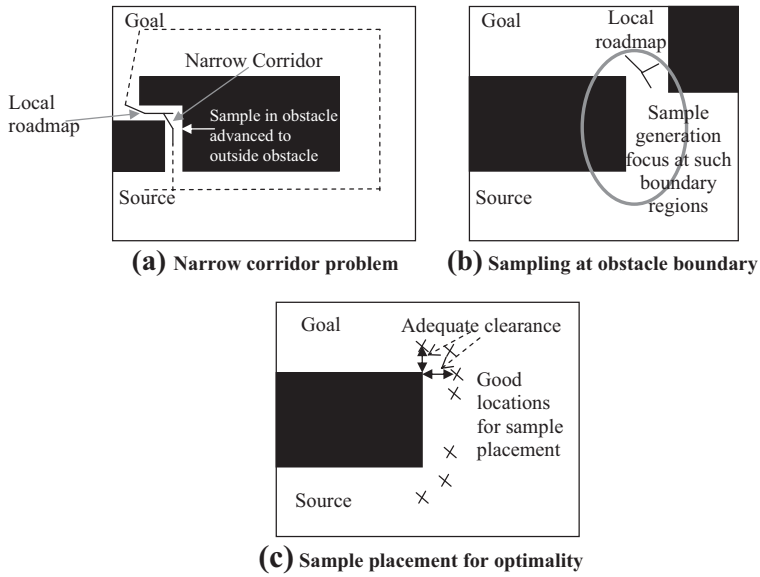
2.3 Problems with Uniform Sampling

The performance of sampling based approaches depends largely upon the quality of samples generated. A typical way is to use a uniform sampling technique where all states in the configuration space are equally likely to be generated. This results in a reasonable roadmap having vertices in roughly all parts of the environment. The uniform sampling approach faces a few problems:

(i) The probability of generation of samples inside a narrow corridor is very low (Fig. 2a). A solution to the problem is to use obstacle based sampling [20] technique wherein a sample inside the obstacle is promoted to the nearby obstacle-free space by travelling in a random direction. Thus a narrow corridor takes much of the samples of the surrounding obstacles. The problem of low probability however still remains if the corridor and surrounding obstacles combined are a small part of the entire environment. A better variant is the uniform obstacle based sampling [21] where the samples are uniformly distributed along the obstacle boundary, independent of the shape of the obstacle. However the method may perform better than bridge-test [22] and Gaussian sampling [23] in cases where obstacles occupy a significant portion of the environment and sampling near the obstacle boundary is not simple, which are also methods for solving the narrow corridor problem. The two approaches [22, 23] also face problems when the narrow corridor is a small part of the entire environment, for which combining them with a uniform sampler is a viable solution [24]. Better sampling techniques more suited for the narrow corridors include Toggle-PRM [25]. Sections 3 and 4 show how the proposed approach builds local roadmaps upon discovery of a narrow corridor so as to almost completely deal with it, without much the need of re-discovery by generation of other samples.
It must be mentioned that obstacle based PRM [20] first attempts simple connections between vertices, followed by more sophisticated mechanisms to connect disjoint vertex sets by edge and node additions. However the author's initial experiments revealed that while it was very easy to have a connected framework between vertices, it was challenging to get redundant edges and useful cycles to discover all homotopic groups. Like in Fig. 2a, challenge was the narrow corridor after the easier path was already obtained. In such a condition the obstacle based PRM reduces to a simple obstacle biased sampling method used in the discussions. The scenario would have been different with only a single narrow corridor as a part of the environment with no possibility of redundant connections, for which finding a feasible connection is a good enough solution.

(ii) It is better to generate more samples around the obstacle boundary and obstacle corners in contrast to the wide open spaces. Every sample becomes a prospective vertex in the path of a robot from its source to its goal, wherein the vertex usually denotes a point of turn for

Fig. 2 Sampling strategy



the robot which it must make in order to avoid some obstacle. If the environment was structured with polygon obstacles all around, keeping samples at the obstacle corners was enough, thereby making a visibility roadmap. While visibility roadmap requires one sample at every corner of the obstacle, random samplers may require more number of samples per corner as samples may be generated near the corner at the boundary face, from where connection to other samples is not easy. Since the obstacles are un-structured and may not have corners, and may be high dimensional, clearly many samples are needed. More samples should be near the obstacle boundaries rather than open spaces. The proposed approach solves the problem by concentrating on obstacle boundaries, but also taking a few samples in wide open space. A local roadmap is generated to completely model a seemingly difficult turn. The concept is shown in Fig. 2b.

(iii) Though the specific problem of focus in this paper is only to create a homotopy conscious roadmap, the location of the individual vertices or samples is important as it contributes towards the generation of optimal paths. A path can be a representative of the path cost of the homotopic group only if the representative path is near-optimal. An ideal location of the sample should hence be that facilitates a short path length and large clearance. Clearance is defined as the smallest distance between the robot and obstacles. Let the robot be at configuration x and occupy a space $R(x)$ in the workspace. Let O be the set of obstacles with o as one of the point obstacles in O. The clearance, denoted by $clearance(x)$ is given by Eq. 4

$$clearance\,(x) = \min_{r \in R(x), o \in O} \|r - o\| \tag{4}$$

Though the optimality of the path does not translate to the optimality of the individual samples, however an understanding of the path optimality can lead to a judicious sample generation. The sample must be around the obstacle boundary, preferably corner (for small path length), and maintain a sufficient distance from the obstacles (for large clearance and high smoothness). The concept is shown in Fig. 2c. Uniform sampling results in a random sample generation and hence the initial few samples lead to a highly sub-optimal sample placement. Similarly approaches biasing sample generation towards the medial axis [26, 27] result in samples being too far from the obstacles for

wide open space scenarios. The proposed paper attempts to generate samples around the obstacle boundary so as to aim small lengths, while being somewhat away to maintain a respectable clearance.

3 Related Works

There has been an interest in being homotopy conscious in the use of PRM. Schmitzberger et al. [28] proposed Homotopy Preserving PRM. They proposed adding edges to a roadmap to make it connected as per visibility, while eliminating any loops to get the least possible paths. The focus of the work was on eliminating and adding edges to get a homotopic structure, while the proposed work focuses on best sampling of the configuration space to get nearly all possible homotopic groups. Of course, the results of the proposed work can be used in the work of Schmitzberger et al. [28] to eliminate excessive paths.

If the environment is structured or semi-structured, the task of homotopy identification is reasonably simpler. Bhattacharya et al. [29] used complex analysis and electromagnetism principles to compute the homotopy and the associated shortest paths. Demyen and Buro [30] used triangulation to get a homotopic structure which was later reduced, and then a search was applied to get the shortest path. Grigoriev and Slissenko [31] took semi-algebraic obstacles in a plane to compute the shortest path within a given homotopic group. The focus of this work is however for unstructured environment.

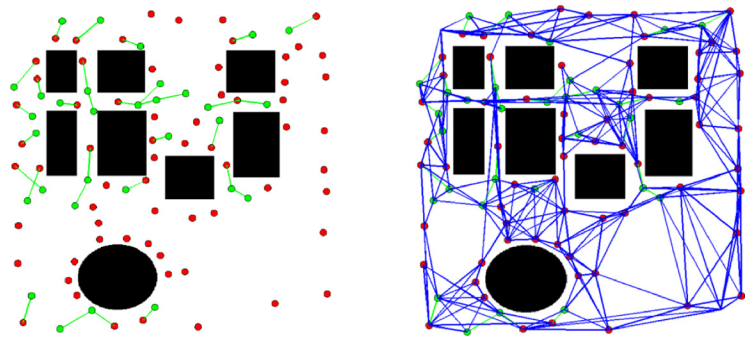
Hsu et al. [32] presented the limitations of the sampling techniques and gave the idea of adaptively combining the sampling techniques. This paper is aimed at a similar notion wherein the sampling design integrates the advantages of various approaches. The notion of adapting between multiple samplers was not taken due to high dependence of the sampler to cost function judging the utility of the sampler, which can be difficult for the problem of homotopy identification. It may further take time for the sampler choice to be adapted. Hence the choice was to make an extensive integrated design instead of adapting the samplers.

Shi et al. [33] further used local roadmaps inside narrow corridors and enabled their connection to the rest of the roadmap using an RRT-like approach. Contrary to the work of Shi et al. [33], the proposed approach does not demarcate corridors as wide and narrow. It assesses the difficulty proportional to the narrowness of the corridor and accordingly deputes samples in the local roadmap. Further it facilitates redundant connections to introduce cycles useful for discovering as many homotopic groups as possible.

Visibility PRM [34] and related techniques explore in a tree-like manner, and hence such approaches are not designed to explore all homotopc groups for which cycles need to be permitted. They however result in generation of spars roadmaps. In another related work Jaillet et al. [35] proposed a transition-based RRT wherein the nodes were first generated. However to be inserted into the tree they must pass the transition test wherein only those nodes were added which do not severely add the traversal cost. In this manner the number of nodes were limited while aiming for paths with better costs. Similarly numerous approaches use spanners [36–38] or related methods to produce roadmaps with limited vertices and edges. The works aimed at eliminating online planning time of the roadmap, while the proposed work is aimed at limiting the roadmap construction time, while still discovering most homotopic groups. Computationally intensive measures cannot hence be taken in the roadmap construction phase.

A significant effort is put to make the RRT optimal and time efficient. Raveh et al. [19] proposed executing multiple RRTs, which were then combined using a dynamic programming based approach. With some additional computational cost, it induces optimality to the solution. Similarly Ferguson and Stenz [39] proposed an iterative format of RRT to make it optimal, wherein the previously generated solutions acted as heuristics to guide the current iteration of RRT towards optimal regions. Kalisiak and van de Panne [40] proposed the use of a flood-fill like mechanism for expansions in RRT called as RRT-Blossom. This was used to bias the expansions towards the goal.

Expansions at critical parts of the environment can be difficult for which Strandberg [41] proposed the use of local trees from various parts of the environment. Expansions of these trees resulted in an insight into handling critical regions of the environment. Urmson and Simmons [42] proposed to use heuristics in the selection of nodes. The heuristics translated to node selection probability of expansion,



(a) Generation of vertices in the proposed approach **(b) Generation of edges in the proposed appraoch**

Fig. 3 Generation of Roadmap **a** Generation of vertices in the proposed approach. The sampled vertices are coloured red. The vertices produced by the local roadmap are colored green. The corresponding edges of the local roadmap are coloured green. **b** Generation of edges in the proposed approach. The edges added between different local roadmaps are coloured blue

and thus the algorithm can be made biased to expand along better paths. The focus of these techniques is to empower node selection and expansion, and not sample generation which is the focus of the proposed work.

4 Algorithm

The algorithm follows the general methodology of PRM [5] wherein the stages are modified to suit the motivations as detailed in Section 2. The general methodology includes the use of a sampler to generate samples which become the vertices of the roadmap (V), computing edges (E) between the roadmap vertices and using the roadmap $G(V, E)$ so formed for query phase which for this paper corresponds to the problem of multi-robot motion planning. Each step is detailed in the following sub-sections. The roadmap for a synthetic test case after the generation of vertices is shown in Fig. 3a and the same roadmap after the generation of edges is shown in Fig. 3b.

4.1 Sampling

The base sampler is a uniform sampler, which uniformly samples the configuration space, $x \leftarrow \mathrm{U}(C)$, where x is a random sample. Since the aim is to focus the samples towards obstacles, the base sampler is modified to produce an obstacle biased sampler given by Eq. 5.

$$x \leftarrow O(C) = \begin{cases} U(C_{obs}) & \text{with probability } \eta \\ U(C) & \text{with probability } 1\text{-}\eta \end{cases} \quad (5)$$

The strategy produces approximately η percent samples in the obstacle prone configuration space C_{obs}. Gaussian [23], bridge [22] and related sampling techniques aim to concentrate on obstacle or obstacle boundaries and hence suggest use of η as 100 %. However such methods may not well sample out environments with wide open free spaces, placing no sample in such areas. Hence the approach uses a large value of η, largely (η percent) focussing on obstacles with a small focus (1-η percent) on open spaces as well.

The obstacle prone samples are then promoted to the nearest point in the free configuration space, given by Eq. 6, which is similar to obstacle based sampling. It is attempted that they further maintain a minimum clearance of C_{min}, in which case they are moved even further by a small value given by Eq. 7.

$$x_{free} = \begin{cases} x + r(\theta) : x + r(\theta) \in C_{free},\ x + r'(\theta) \notin C_{free} \forall 0 \le r' < r,\ x \notin C_{free} \\ x \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad x \in C_{free} \end{cases} \quad (6)$$

$$y = x_{free} + c(\theta) :$$
$$c = \arg \max_{0 \le c' \le C_{\mathrm{min}}} Clearance\left(x_{free} + c'(\theta)\right) \quad (7)$$

Here x_{free} is sample promoted to the free configuration space (if needed) and y is the sample which further maintains a minimum clearance of C_{min}. θ is the direction of movement of the point, $r(\theta)$ denotes a vector of magnitude r and direction θ. $Clearance(x)$ is the clearance function which returns the distance from the configuration x to the closest obstacle.

The number of vertices ($|V|$) in the roadmap need to be limited so as to invest the least computation in

the roadmap construction phase (and also the query phase). The best way to do so is to avoid any unnecessary cycles in the graph, which can be approximately checked. However such approximate computations incur costs equivalent to multiple edge collision checks and can be used only if query computational time is to be reduced. It is better to add such vertices, investing a part of cycle checking time to connecting edges between the vertices, thereby adding query computation time but saving on the roadmap construction time. However two vertices very close to each other are clearly useless and this can be checked in a small time. Samples are taken iteratively and added to the roadmap as vertices. Any such sample (y) is admissible as a vertex only if it lies at a distance of more than $d_{\min}$ from the earlier existing vertices given by Eq. 8.

$$\min_{v \in V} \|v - y\| > d_{\min} \tag{8}$$

4.2 Local Roadmaps

The samples generated can be found anywhere in the free configuration space including regions of narrow corridors, situations requiring tight steering corrections, wide open spaces, and obstacle boundaries with wide open space at the other side. The first two situations are hard to handle for sampling based motion planners, the first more than the second. The steps taken in the literature for handling the situation have been taken into account in the algorithm design properties noted in Section 4.1. However getting a sample inside a narrow corridor can be valuable when the narrow corridor is a small part of the entire environment. Approaches in the literature go forth with only adding the detected sample in the narrow corridor. However it is proposed here to generate a *small local roadmap* around the location of the sample in the narrow corridor, thus enabling the algorithm to almost completely deal with the corridor when it is first discovered, rather than waiting for the algorithm to be lucky in finding another sample in the corridor later.

In order to test whether the sample is inside a narrow corridor, a generalized bridge test [22] is proposed. The first generalization made is that the check is done in all (sampled) directions. The second generalization is that it is not only important whether the corridor is narrow or not, but also how narrow the corridor is. The check is given by Eq. 9.

$$D(y) = \min_{\theta}(Clearance(y, \theta) + Clearance(y, \pi + \theta)) \tag{9}$$

Here $Clearance\ (y, \theta)$ denotes the clearance at point y in direction θ. $\pi +\ \ \theta$ is the angle directly opposite to θ. $D(y)$ denotes the width of the corridor that contains the sample y.

Let the local roadmap produced by the discovery of a narrow corridor by the sample y be $G^y(V^y, E^y)$. The total number of vertices that this roadmap may have $|V^y|$may be given by any function $f(D)$ monotonically decreasing with the width of the corridor, implying that a narrower corridor is harder to deal with and will hence have a larger size of the local roadmap. The size is subjected to a maximum threshold ($V^y_{\max}$) to avoid reserving too many nodes for the local roadmap at the cost of the global roadmap, while a minimum threshold of 1 naturally applies. A simple way to set the function is to take a number of scenarios, calculate the widths at key locations, and decide the number of nodes that would be apt. A function can be easily regressed from the data. Since the maximum threshold is fixed and the output has to be rounded to integer values, there are a few regions and a small set of values to regress. Currently the function is taken as inverse of the width as given in Eq. 10.

$$\left|V^y\right| = f(D) = \frac{\delta}{D(y)} \tag{10}$$

Here δ is an algorithm parameter that relates the width to the number of vertices in the local roadmap.

To produce the local roadmap, starting from the sample y, an RRT style exploration [15, 16] with a variable step size is used. At any iteration a random sample from the local roadmap so far is taken, z'=*rand*(V^y). Here *rand* returns a random sample from the set taken from a uniform distribution. A ran-

dom sample (z) is produced in the close vicinity of z'. z is added as a vertex if (a) it is collision free, $z \in C_{\text{free}}$. (b) The edge from z' to z is collision free, $\exists e{:}e(t) \in C_{\text{free}} \forall 0 \leq t \leq 1$, $e(0) = z'$, $e(1) = z$. Currently only straight lines are considered. (c) The minimum distance criterion to the other vertices as mentioned in Eq. 8 is followed, which includes all the vertices added to any local (or global) roadmap. If z is admissible as a vertex of the local roadmap, the corresponding edge is added to E^{y}.

The expansion is stopped when the required number of vertices (with edges) have been produced in the local roadmap, or a maximum number of failed attempts have been reached. The maximum failed attempts naturally depends upon the total number of vertices to be added as narrower corridors require more failed attempts for each insertion of a vertex. Figure 2a shows the local roadmaps produced on a synthetic map. The red coloured nodes are the samples generated. If a sample is in a narrow region, the local roadmap is generated. The local roadmap vertices and edges are shown in green.

Based on the discussions so far, it can alternatively be stated that the global roadmap $\varsigma(V, E)$ is such that each vertex $v \in V$ is a local roadmap $G^{\text{v}}(V^{\text{v}}, E^{\text{v}})$, and each edge $e \in E$corresponds to connections between two local roadmaps $e{<}v_1, v_2{>}$: $v_1 \in V^1$, $v_2 \in V^2 \neq V^1$. $|V|$denotes the total number of disjoint local roadmaps. The task of production of vertices of the global roadmap or production of local roadmaps was done in this sub-section.

4.3 Computing Edges

The next task is to connect *bridge edges* that connect two disjoint local roadmaps and serve as edges of the global roadmap. Two connection schemes are used: fully-connected and k-connected.

A fully-connected roadmap strategy seeks to find connections between any two pair of vertices (say v_{i} and v_{j}). If a collision-free connection exists ($\exists e$: $e(t) \in C_{\text{free}} \forall 0 \leq t \leq 1$, $e(0) = v_{\text{i}}$, $e(1) = v_{\text{j}}$), the corresponding edge is added to the graph. Here $e(t)$ represents any general point in the edge from v_i at t=0 to v_j at t =1. Currently only straight line connections are checked ($e(t) = v_{\text{i}} + t.(v_{\text{j}}\text{-}v_{\text{i}})$).

The feasibility checking for any edge [43] is first done by a *lazy collision checker* ($e(t) \in C_{\text{free}} \forall t \in \text{U}^{\alpha}(0,1)$). Here $\text{U}^{\alpha}(0,1)$ denotes α uniformly sampled points in the interval of 0 and 1. α corresponds to the granularity of collision checking. The method checks for collision possibility across α sampled points, followed by a complete collision checker ($e(t) \in C_{\text{free}} \forall 0 \leq t \leq 1$, $e(0) = v_{\text{i}}$, $e(1) = v_{\text{j}}$). A lazy collision checking was not applied in Section 4.3 since it was known that the vertices would be reasonably near.

A fully connected roadmap incurs significant computational costs. Hence a k-connected variant of the algorithm is proposed, where an edge can have a maximum of k connections while the connections are made with the priority of the nearest neighbours. Because of the choice of the sampling technique, sometimes it may be impossible to make connections between the nearest samples with a need to consider the further samples, still the number of connections should not exceed k (say the sample is near an obstacle boundary, outside one face, with most samples or a local roadmap at the other face)

While a naïve implementation of the connection strategy is possible, it is intended to further use the property of a *fully connected local roadmap* (directly or indirectly through other vertices) to speed up the computation. Although indirect connections are not counted in designating a roadmap as fully connected, it is done here because the local roadmap spans across a very limited volume and reaching a vertex directly or through some other vertex does not have a large difference in path costs. If a vertex v is connected to any vertex v_{j}, it may hence be assumed that it is also (indirectly) connected to all the vertices in the local roadmap of v_{j} (say v^{z}).

Let $\kappa(v)$ denote the number of connections maintained by the vertex v at any time. As per the above discussions that includes all the vertices in the local roadmap in which v is placed (say V^{y}: $v \in V^{\text{y}}$), as well as all vertices of any local roadmap to which the vertex v has a direct edge to (or all vertices of local roadmap V^{z}whose any member maintains an edge with v). $\kappa(v)$ may be given by Eq. 11. Here the edge function E is taken different from the function κ since E stores only direct edges between the vertices as a formal definition of the roadmap, while κ stores the direct and indirect edges.

$$\kappa(v) = \left| \left(V^{y} - v\right) \cup \bigcup_{z \neq y, \langle v, v_j \rangle \in E, v_j \in V^{z}} V^{z} \right| \tag{11}$$

Edges are added iteratively in an initially completely disconnected roadmap obtained from Section 4.3, barring the local roadmap edges. For any vertex v, an edge is attempted to be added with any vertex v_i, such that: (a) $v \neq v_\mathrm{i}$, (b) v_i is the closest disconnected neighbour for which edge feasibility checking has not already taken place, (c) v is not already connected to the fullest extent possible, or $\kappa(v) < k$. (d) v and v_i are not in the same local roadmap in which case they are already assumed to be directly or indirectly connected, or $v \in V^\mathrm{y} \wedge v_\mathrm{i} \notin V^\mathrm{y}$, (e) the local roadmap $G^\mathrm{z}(V^\mathrm{z}, E^\mathrm{z})$ to which v_i belongs is not already indirectly connected to v through a sister vertex v_j, or v is not already connected to a vertex v_j such that v_j and v_i are in the same local roadmap, or

$$v_i \notin \bigcup_{z \neq y, \langle v, v_j \rangle \in E, v_j \in V^z} V^z.$$

The collision checking happens using a lazy collision checking followed by a complete collision checking. The addition of edges in the map is shown in Fig. 2. The edges added are coloured as blue to differentiate them from the local roadmap edges. The algorithm of computing the vertices is given by Algorithm 1. The algorithm to compute the edges is given by Algorithm 2. The set P in the algorithm stores all elements iteratively used for condition (e) above. The algorithm is for the k-connected variant, while the fully connected variant is the simpler case that can be easily derived from Algorithm 1 and Algorithm 2.

Algorithm 1 is used to compute the vertices of the roadmap. A sample is generated using an obstacle biased strategy (Line 3), which is propagated to the obstacle boundary (Line 4), and still further to maintain a preferred clearance, if available (Line 5). If the sample is away from the pre-existing samples in the roadmap (Line 6), it needs to be added. A small local roadmap may be added as per the difficulty of the region. The corridor width is assessed (Line 7) to know the number of samples to produce in the local roadmap (Line 8). A local roadmap is initialized with the newly generated node alone (Line 9). A random vertex is selected from this roadmap (Line 11), which is extended in a random direction (Line 12) by a random step size (Line 13) to generate a new node (Line 14), which on meeting the acceptance conditions of non-collision of vertex and edge and not pre-existing in the roadmap (Line 15), is added to the local roadmap by the addition of vertex (Line 16) and edge (Line 17). All local roadmaps are added to the roadmap, along with the vertices (Line 20) and edges (Line 21). The list of connected vertices for each vertex is kept as the vertices in each local roadmap (Line 22).

Algorithm 2 adds the edges in the roadmap. For every vertex v (Line 1), first the disjoint vertex set or the local roadmap is computed to which the vertex belongs (Line 2). Then iteratively edge connection attempts are made between the selected vertex (v) and the other vertices (v_i) in the increasing order of distance between the vertices (Line 4). An edge can only be added till maximum connectivity is not reached (Line 5) and if the two vertices are not directly or indirectly connected, in other words, the two vertices do not belong to the same local roadmap and do not belong to local roadmaps connected to each other (Line 6). The local roadmap of v_i is computed (Line 7). A straight line connection is attempted (Line 8). If the connection is collision-free (Line 9), the corresponding bi-directional edge is added (Line 10). The vertex v is said to be connected to all vertices in the local roadmap of v_i and vice versa, and hence their connectivity is increased (Line 12 and 13). The set P maintains all connections between the local roadmap V^y and other local roadmaps as they are made. It is initialized by a NULL (Line 3) and upon addition of any local roadmap, the set is updated (Line 11).

4.4 Multi-Robot Path Planning

The focus of the paper was construction of a roadmap in order to represent nearly all possible homotopic groups, which can be used to query for robotic tasks. Although the focus of the paper is not multi-robot path planning, the same was an inspiration of the work, and is hence dealt with in this sub-section only for the sake of completeness of the work.

Priority based planning [44] is used, wherein the robots are given priorities and planned strictly in the same order. The priority assignments can be optimized [45] if the query computational time is small. Prioritized A* algorithm is used for planning. The A* algorithm uses the heuristic function as the Euclidian distance to the goal upon the maximum speed, which is the earliest that the robot can expect to reach the goal. The historic function is the time from the source which is edgewise discounted by a penalty function for lack of maintenance of a high clearance.

Algorithm 1 ComputeVertices(C, {s_r, % g_r})

Line 1 G<V,E> ← <{s_r, g_r},∅>
Line 2 While Not Stopping Criterion
Line 3 x ← random sample from obstacle biased sampler (5)
Line 4 x_{free} ← sample from motion of x to nearest C_{free} (6)
Line 5 y ← sample in C_{free} maintaining preferred clearance C_{min} (7)
Line 6 if $\min_{v \in V} \|v - y\| > d_{min}$(8) [Note that v iterates through all vertices of all member local roadmaps]
Line 7 D(y)←Corridor width (9)
Line 8 M ←maximum vertices in G^y (10)
Line 9 G^y<V^y,E^y> ← <y, ∅>
Line 10 while $|V^y|$<M or maximum failed iterations
Line 11 $z' \leftarrow$rand(V^y)
Line 12 $\theta \leftarrow$ random direction vector
Line 13 r ← rand(min step size, max step size)
Line 14 z ← $z' + r(\theta)$
Line 15 if $\left(\begin{array}{l} \left(z \in C_{free}\right) \wedge \left(\exists \text{e:e(t)} \in C_{free}, 0 \leq \text{t} \leq 1, \text{e}(0) = \text{z}', \text{e}(1) = \text{z}\right) \\ \wedge \min_{v \in V \cup V^y} \|v - z\| > d_{min} \end{array} \right)$
Line 16 $V^y \leftarrow V^y \cup$z
Line 17 $E^y \leftarrow E^y \cup$<z,z′>∪<z′,z>
Line 18 end if
Line 19 end while
Line 20 V←V∪V^y
Line 21 E←E∪E^y
Line 22 κ(v)←$|V^y|\forall v \in V^y$
Line 23 end if
Line 24 end while
Line 25 return G<V,E>

The A* algorithm may prefer wide open spaces in contrast to narrow passages computed in the roadmap construction. It must be noted here that even if the roadmap generation method is complete and guarantees the generation of all possible homotopic groups, prioritized A* algorithm may still not be complete. Prioritization can reduce completeness as it does not model plans wherein multiple robots mutually adjust their paths, without any one robot taking a complete priority over the other. The priority assignment can also be made dynamic as displayed in the work of Clark et al. [46], who used kinodynamic single-query motion planning for navigating multiple mobile robots, using a dynamic prioritization technique.

Let there be N robots in the system, each with source s_r and goal g_r. The roadmap is built with a static configuration space C^{static}, assuming no mobile robots. The robots are prioritized. Let the priority of any general robot be p_r. The robots are planned in the order of priority and at any state of planning of the robot r, the trajectory of any robot q(say τ_q) is known if $p_q < p_r$. A dynamic configuration space for any general time t is hence given by Eq. 12.

$$C_{free}^{dynamic}(t) = C_{free}^{static} \setminus \bigcup_{p_q < p_r} x : collides\left(\tau_q(t), x_r\right) \tag{12}$$

To check collisions a mapping to the workspace is done. The function $collides(\tau_q(t), s)$ checks if the robot q placed at a point $\tau_q(t)$ collides with the robot r placed at a point x.

The same space is additionally used for collision checking for all edges of the roadmap. Using the criterion, let the path computed in the query phase of

Algorithm 2 Compute Edges(C, G<V,E>)

```
for all v∈V
    y: v∈V^y
    P← ∅
    for all v_i(≠v)∈V in increasing order of ||v–v_i||such that <v,v_i>∈E
        if κ(v) ≥k, break, end if
        if v_i ∉P∧v_i ∉V^y
            z: v_i ∈V^z
            e(t)=v+t.(v_i-v)∀0 ≤t≤1
            if(e(t)∈C_free∀t∈U^α(0,1)) ∧(e(t) ∈C_free∀0 ≤t≤1)
                E←E∪<v,v_i>∪<v_i,v>
                P ← P∪V^z
                κ(v) ← κ(v)+|V^z|
                κ(v_i) ← κ(v_i)+|V^y|
            end if
        end if
    end for
end for
return G<V,E>
```

the roadmap be $Path(s_r, g_r, C_{free}^{dynamic})$. The path is further post-processed by using simple operators like deletion of points for path shortening and locally optimizing the points for path smoothening, along with the use of spline curves for path smoothening. The post processed path is traversed with uniform speed which becomes the robot's trajectory τ_r.

5 Results

The algorithm was tested via a number of simulations. Initial testing involved generating a roadmap over different scenarios. This was used for the problem of motion planning, while both the roadmaps and the paths were displayed. The robot considered was small and rectangular in shape and hence the configuration space was SE (3), which has 3 components, the position (x, y) and the orientation in SO (2), which is basically the orientation angle (θ). The configuration space was mapped on the 2D figure with an R^2 configuration space, by eliminating the orientation angle in the plot. The map was taken of the size 500×500 while the size of the robot was taken as 10×10. The roadmap was used to assess the working of the algorithm. The metric used to judge the performance is the additional number of edges required that lead to representation of all homotopic groups (3). While the algorithm works autonomously for any scenario, the metric needs to be computed manually due to no existent algorithm for the purpose. It is however easy to compute the metric for an almost fully connected roadmap. Automated methods could be possible with the assumption of a structured environment, wherein all possible homotopic groups can be easily computed using a Voronoi roadmap and listed using a typical graph search, and further using algorithms to check if two paths belong to the same homotopic groups. For an unstructured environment this can only be done on coarser resolutions. Since presence of a narrow corridor is a challenge, the resolution of homotopy checking cannot be kept low. Since the test metric was not the specific focus of the work and was not needed in the general working of the algorithm, a manual method was preferred over researching over algorithms for automated checking.

The algorithm has little parameters which significantly affect the algorithm performance and are of practical value. The algorithm does not add any special parameter of relevance, not present in PRM or other popular approach in robot planning. The stopping criterion is fixed as the total number of vertices and varies as per the scenario as experimented and set

below. k is taken to be 10 to avoid an overly-connected graph. η can be fixed to any convenient large value, which is taken as 0.9. δ is taken as 100, while the number of vertices are limited to lie between 1 and 5. Minimum distance allowed between vertices can be fixed to any convenient small value, taken as 25 for experiments. The minimum clearance is a design parameter and does not affect this algorithm performance, which is fixed to a very small value of 5. The factor α for lazy collision checking is also not directly related to the algorithm and is kept such that samples at a distance of 40 are generated. The factor was experimentally studied in [43].

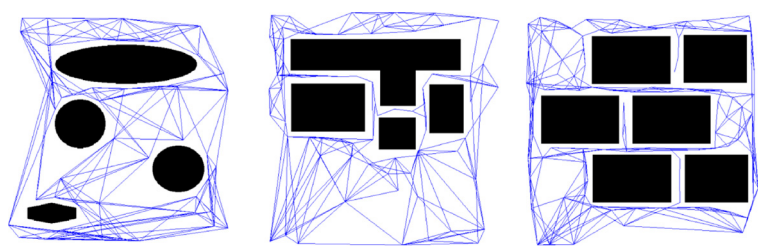
The performance of the algorithm is judged against a standard implementation of PRM using a uniform sampler and a PRM using an obstacle based sampling technique [20]. Note that Gaussian [23] and bridge test [22] were excellent algorithms against which the performance could be tested. However obvious limitations on scenarios as noted in Section 2.3, as opposed to the requirement of performance over a wide variety of scenarios, discouraged their use. Further manual metric computation limited the number of algorithms that could be tested and only the most competing ones were retained.

Three scenarios are discussed here. Note that in all experiments, due to the size of the robot, the widths shown in the environment are smaller than the ones available. The first scenario mostly consisted of wide open spaces with some obstacles around. However there was a small region between two obstacles where the space was less. Further, for too less samples, it may be difficult to connect between obstacle corners and the environment boundary. The aim behind the scenario was to test if all homotopic groups can be discovered for a very small number of vertices, yielding roadmap construction in small computational times. The roadmap constructed with only 50 samples by the algorithm is shown in Fig. 4a.

The second scenario had 3 small narrow corridors which had to be discovered while the rest of the environment had wide open spaces. All the three corridors should be found as well as connected to each other for a success. The scenario is shown in Fig. 4b with the performance of the algorithm with 75 samples. The last scenario consisted of multiple narrow corridors with barely any open space with the rest of the environment. Each of these corridors had to be connected to all the neighbouring ones, while some space was available for the connections to happen. The environment and the performance of the algorithm for 100 samples is shown in Fig. 4c. It can be seen that while the algorithm succeeded in discovering all the corridors, it was unable to connect one of those and hence connection is an equally important problem as discovering.

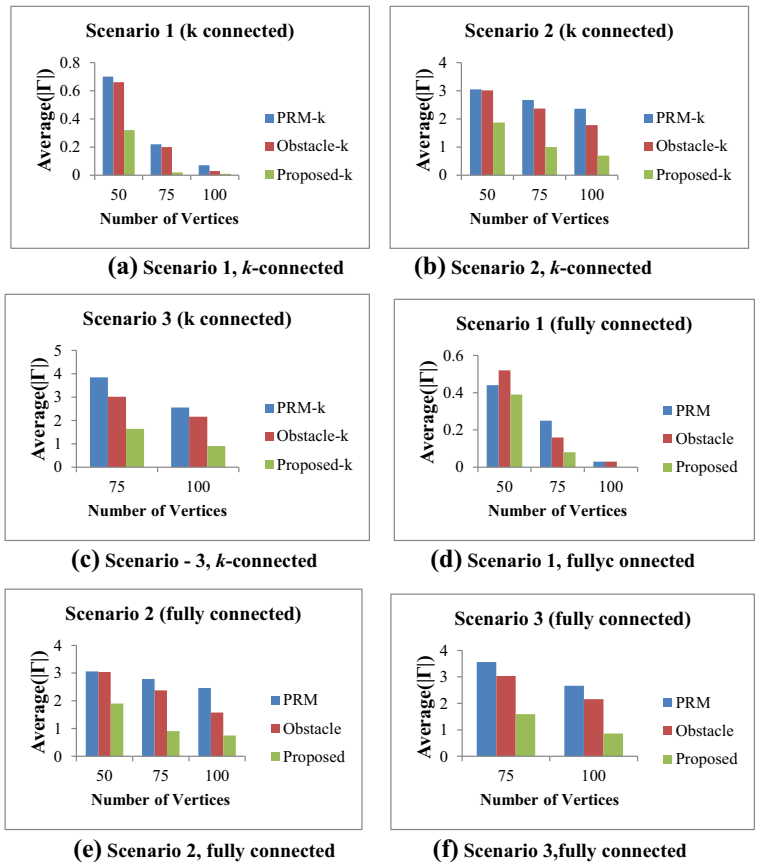
The stopping criterion of the algorithm was kept as the total number of vertices in the roadmap $|V|$. The performance against a number of limiting values was tested. Both the fully connected and k-connected variants were tested separately. The performance was tested over 100 cases and the results were averaged. The computation time in each of the cases was also noted. The results noting the algorithm performance are given in Fig. 5 and the results giving the computational time are given in Fig. 6. For the last scenario testing was not done with 50 samples as they were clearly insufficient to discover all the corridors in the scenario.

Figure 5a, b and c show the results for the three scenarios for the k-connected variant while Fig. 5d, e and f show the results for the fully connected variant. It can be easily seen that the proposed approach performs best for all the scenarios followed by the obstacle based sampling, while the uniform sampling approach performs the worst. The results are an experimental conformation to the fact that the proposed approach can adjust to the different types of scenarios, very



(a) Scenario 1 **(b)** Scenario 2 **(c)** Scenario 3

Fig. 4 Maps and roadmaps for various scenarios



(a) Scenario 1, *k*-connected **(b)** Scenario 2, *k*-connected

(c) Scenario - 3, *k*-connected **(d)** Scenario 1, fullyc onnected

(e) Scenario 2, fully connected **(f)** Scenario 3,fully connected

Fig. 5 Performance of the algorithms

quickly resulting in discovering most of the homotopic groups. While the easier and relatively challenging homotopic groups may be discovered in the initial very few samples, the most challenging ones require only some additional iterations.

Figure 6 shows a comparison of the computation time for all the scenarios. For the first scenario having open spaces, shown in Fig. 6a and d, all approaches seem to be equally computationally expensive, which is an expected trend. Since the environment mostly has wide open spaces, there is a minimal computational effort spent in advancing an obstacle-prone configuration to the nearest boundary. The results, shown in Fig. 6b and e for the second scenario and Fig. 6c and e for the third scenario, show that the proposed approach has consistently less computational expense. The uniform sampling PRM approach wastes a significant time in finding a collision-free sample. The obstacle based sampling approach saves that time by advancing a collision-prone sampling to the collision-free area, while such an advancing also takes time. The proposed approach also saves the obstacle-free sample generation time by advancement of the sample, while investing time on advancement of the sample. However once such an advancement has been made, the approach gains by generation of multiple connected samples in the nearby area, which are computationally easy to find. Thus the generation of a local roadmap is a boon for performance not only for the metric of study, but also for reducing the computation time.

For the sake of completeness, the results of planning multiple robots on the produced roadmap are also presented. In each of the scenarios two robots were taken at opposite corners, which had to reach the other opposite corner. Since the robots have complementary goals and sources and use the same roadmap, ideally they would get the same optimal path, disregarding the other robot. However, using priority based A* algorithm, the second robot knows the path of the first, and may hence aim to track a different homotopic group, or may aim to modify its path in the same homotopic

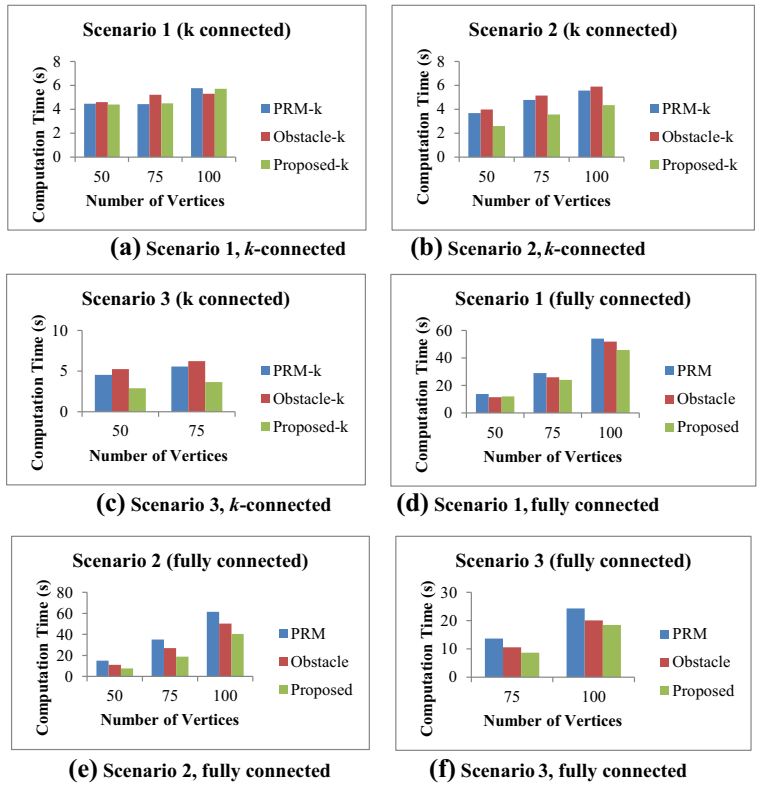


Fig. 6 Computation times of the algorithms

group. Both the cases were seen. The post processing algorithm attempted to maintain a clearance larger than the minimal used in the roadmap computation. The results for different scenarios are given in Fig. 7. The time instance when the robots avoided each other is given for each scenario, followed by the complete path at the end.

6 Discussions

The algorithms are assessed from the point of view of computational time. The complexity of the method is difficult to understand, as most of the terms otherwise stated as constant also need to be incorporated to best understand the algorithm. If these terms are not incorporated, the complexities of the algorithm will come out to be the same. In this section the computational time of the methods is studied from a complexity perspective. Let us define some terms first:

Vertices

n: The number of vertices in the roadmap

p: The probability of a random sample to be in free space given by $|C^{free}|/|C|$

$\bar{p}$: The probability of a random sample to be obstacle prone given by $|C^{obs}|/|C|$

Edges

k: The number of vertices considered for edge connectivity

$||E||$: The average length of edge between any two vertices

D : The maximum obstacle length

Proposed Method

S: The maximum step size for a local edge

η : The ratio of samples generated from obstacle based sampling

W: Maximum width of the corridor considered

γ : Average number of vertices in every local roadmap

n/γ : Number of local roadmaps (n/γ roadmaps each with average γ vertices make a complete roadmap of size n)

k/γ -1 or n/γ -1: The number of local roadmaps that each local roadmap is connected to (Every vertex

is connected to γ vertices in its own local roadmap and k/γ -1 roadmaps, each with γ vertices, totalling k vertices ($\gamma + (k/\gamma$ -1). γ) for a k-connected roadmap) and n vertices ($\gamma + (n/\gamma$ -1). γ) for a fully-connected roadmap)

The complexities of the different algorithms are as under:

PRM

Vertex: O(n.(1+p)) where the term (1+p) is to discard samples generated in obstacles.

Edge (fully-connected): O(n^2.$||E||$) to attempt connections between every combination of vertices with an average edge length as $||E||$.

Edge (k-connected): O(nk.$||E||$) with a similar reasoning

Obstacle-based

Vertex: O(n.(1+$\bar{p}$).D) where the term (1+$\bar{p}$) is to discard samples generated in free space, while every successfully generated sample needs to travel a distance of D to reach the obstacle boundary with a constant clearance.

Edge (fully-connected): O(n^2.$||E||$)

Edge (k-connected): O(nk.$||E||$)

Proposed

Vertex Generation: The total number of vertices generated by the obstacle biased sampling methods are only $(n - \gamma)$ rather than n, because γ vertices are produced by the local roadmap generation method. Out of these $\eta(n - \gamma)$ vertices are produced by the obstacle based sampling method while (1-η).($n - \gamma$)vertices are produced by the uniform sampling method. Noting the complexities above, the complexity to generate the samples is:

$$\text{Vertex generation complexity} = \mathrm{O}(\eta(n-\gamma)\,.\,(1+\bar{p})\,.\,(D+W) + (1-\eta)\,.\,(n-\gamma)\,.\,(1+p)) + \mathrm{O}(\text{vertex generation of roadmap})$$

The term W is added to the first term as an additional step is performed to find the width of the corridor. The complexity of generation of local roadmap accounts for generation of random sample in the local vicinity and checking for the local edge, which gives the complexity as O(γS). Thus the resulting complexity is:

$$\text{Vertex generation complexity} = \mathrm{O}(\eta.(n-\gamma)\,.\,(1+\bar{p})\,.\,(D+W) + (1-\eta)\,.\,(n-\gamma)\,.\,(1+p)) + \mathrm{O}(\gamma.S) \approx \mathrm{O}((n-\gamma)\,.\,(1+p)\,.\,(D+W))$$

Edge Connection (fully connected): Every vertex is already connected to γ vertices directly as a result of the local roadmaps made earlier. Every local roadmap will be connected to (n/γ-1) vertices directly and thus connecting to γ vertices indirectly in each of those roadmaps. The local roadmap edges were made during vertex generation and hence its complexity is not added. Only

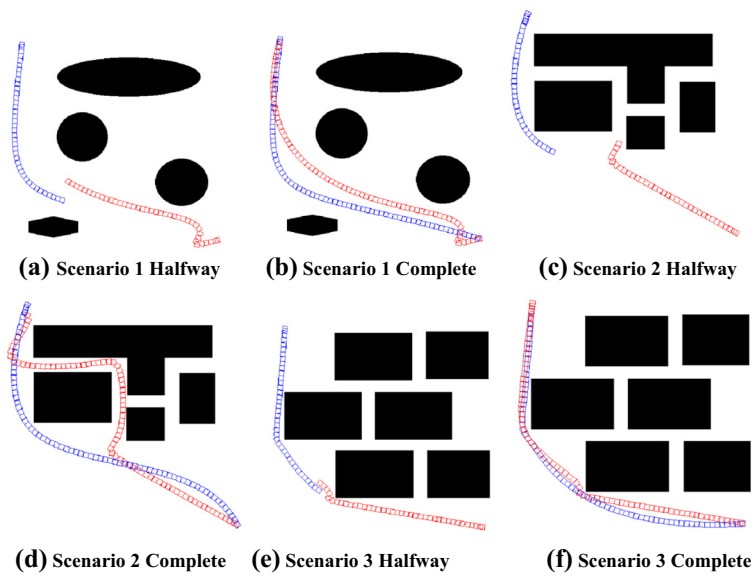


Fig. 7 Simulation Results for Multi-Robot Path Planning

connections between the local roadmaps are made. Hence the complexity is O($(n/\gamma$-1).$||E||$)

Edge Connection (k-connected): O($(k/\gamma$-1).$||E||$)

The results are summarized in Table 1.

Based on Table 1 and Fig. 6, let us first analyze the fully-connected variant. Obstacle-based sampling generally spends more time in vertex generation as compared to PRM with uniform sampling, however in both methods the major cost is of edge connectivity. Because vertices in obstacle based sampling are near the obstacle boundary, the average edge length ||E||in obstacle based sampling is smaller which overcomes the extra time spent in vertex generation. Similarly the proposed method clearly has a larger complexity of vertex generation, however a smaller complexity of edge connection. This is because some of the edges are locally connected by small lengths. Also indirect connections between edge of one local roadmap and another local roadmap are said to be (indirectly) connected (if the two local roadmaps are connected) without adding a direct edge or checking for a connection. This saves computation time.

The same reasons hold for k-connected variants with the only difference that in k-connected variants, because only nearest vertices are checked for connectivity, the average edge length of obstacle based sampling may not always be shorter than that of uniform sampling PRM. Hence the computational time of obstacle based sampling method may be higher due to larger time spent in the computation of vertices.

The next major issue is to prove the completeness of the algorithm. By definition, a deformation retract guarantees completeness while also holding the property of generating all homotopic groups. The proof is largely inspired by the notion of a deformation retract. To prove completeness of the constructed roadmap, essentially the notions of accessibility (it is possible to go from the source to some point on the roadmap), departability (it is possible to leave the roadmap at some point and go to the goal) and connectivity (the point of accessing the roadmap and the point of departing the roadmap are connected to each other) need to be ascertained. We make use of the notion of homology of paths to prove these properties as most of the discussions of homotopy in this paper are also valid for homology. Suppose a roadmap $G(V, E)$ exists that guarantees representation of all homotopic and homologous groups. Let s and g be any pair of source and goal in the configuration space. To prove accessibility, let us first assume that the vertex s is not visible to any vertex or edge in G, or simply that it does not satisfy the property of accessibility. In such case consider a path $\tau(V_1, s)$ from V_1 ($\in V$) to s and another path $\tau(s, V_2)$ from s to V_2 ($\in V$). Consider the path $\tau(V_1, s, V_2)$ from V_1 to V_2 through s, formed by appending the two paths $\tau(V_1, s)$ and $\tau(s, V_2)$ one after the other. Since the roadmap guarantees representation of all homologies $\tau(V_1, S, V_2)$ is in some homologous group as some other path $\tau(V_1, V_2)$ represented in the roadmap. Hence the area formed by $\tau(V_1, S, V_2)$ and the inverse of $\tau(V_1, V_2)$ from V_2 to V_1 is collision-free. By this definition s can be connected to some point x in $\tau(V_1, V_2)$ by a straight line without encountering obstacles, which nullifies the notion of the roadmap being non-accessible. Similarly the notion of departability can be proved with some line leading from a point y in the roadmap to the goal g.

To prove connectivity, first assume that no path exists between x ($\in V$) and y($\in V$) or that the property of connectivity does not exist. This can only happen when the roadmap consists of at least 2 subgraphs, one containing x and another one containing y. Consider that a path $\tau(x, y)$ between x and y exists and is not represented in the roadmap. A path $\tau_G(x, y)$

Table 1 Algorithm Complexity Analysis

S. No.	Algorithm	Vertex Generation	Edge Generation	Total
1.	PRM (k-connected)	O(n.(1+p))	O(nk.$\|\|E\|\|$)	O(nk.$\|\|E\|\|$)
2.	Obstacle-based (k-connected)	O(n.(1+$\bar{p}$).D)	O(nk.$\|\|E\|\|$)	O(nk.$\|\|E\|\|$)
3.	Proposed (k-connected)	O(($n-\gamma$).(1+p).($D+W$))	O(n.(k/γ-1).$\|\|E\|\|$)	O(n.(k/γ-1).$\|\|E\|\|$)
4.	PRM (fully connected)	O(n.(1+p))	O(n^2.$\|\|E\|\|$)	O(n^2.$\|\|E\|\|$)
5.	Obstacle-based (fully connected)	O(n.(1+$\bar{p}$).D)	O(n^2.$\|\|E\|\|$)	O(n^2.$\|\|E\|\|$)
6.	Proposed (fully connected)	O(($n-\gamma$).(1+p).($D+W$))	O(n.(n/γ-1).$\|\|E\|\|$)	O(n.(n/γ-1).$\|\|E\|\|$)

must exist in the roadmap which is homotopic to this path, otherwise the guarantee of representation of all homotopic groups is not satisfied. For this x and y must belong to the same sub-graph or that the connectivity property must hold.

Being built over PRM, the algorithm is probabilistically complete. The notion of optimality can be broken down into the ability to find the optimal homotopic group and subsequently to compute the optimal path within the homotopic group. The necessity of any optimal algorithm is hence representation of all homotopic groups, which is the property considered in the roadmap. The notion of optimality is largely related to the use of path length as the cost function with a little consideration of clearance. In such a case sampling near the obstacle boundaries makes the algorithm generate better paths. As the approach does not compute paths, rather concentrates on roadmaps alone, the notion of optimality is discussed rather informally.

7 Conclusions

The paper studied the problem of making a roadmap generation method using PRM such that most homotopic groups are represented in small computation times. A new sampling technique was proposed that preferred obstacle boundaries, created small local roadmaps on identification of narrow corridors and preferred to maintain a small local clearance. An edge connection strategy was devised which discounted for the number of connection attempts by assuming that if two local roadmaps are connected by an edge, all pairs of vertices between the two local roadmaps are also said to be indirectly connected without actually attempting the connections. The approach was tested for a number of scenarios ranging from open spaces to multiple narrow corridors. The results revealed that the approach performed better than popular approaches, while also taking lesser time of computation.

The immediate next future work of the paper is to use the proposed algorithm for the problem of multi-robot path planning. The author aims to use the previously published work ([2] and its extension which is currently under review) for the same. The extension requires calculation of paths in multiple competitive homotopic groups, execution of the earlier algorithm across all homotopic pairs, decision of execution time investment in each pair to have a quick idea of the tentative computation cost, finally selecting a homotopic group combination, and computation of the final plan. Each step has some key research challenges to address which would be dealt with in the future.

Further the process of judging the performance metric needs to be automated. A major problem with the performance metric is that it rewards the ability to find connections inside the narrow corridors, but does not partially reward incomplete connections depending upon the portion of the corridor covered. Such a metric would have resulted in greater insights. Finally the algorithm needs to be expanded to scenarios of incomplete environment information, information shared across the robots, computation shared across the robots and parallel processing for enhanced offboard computation.

References

1. Kala, R.: Rapidly-exploring Random Graphs: Motion Planning of Multiple Mobile Robots. Adv. Rob **27**(14), 1113–1122 (2013)
2. Kala, R.: Coordination in Navigation of Multiple Mobile Robots. Cybern. Syst **45**(1), 1–24 (2014)
3. Kala, R., Warwick, K.: Multi-Level Planning for Semi-Autonomous Vehicles in Traffic Scenarios based on Separation Maximization. J. Intell. Rob. Syst **72**(3-4), 559–590 (2013)
4. Kala, R., Warwick, K.: Planning Autonomous Vehicles in the Absence of Speed Lanes using an Elastic Strip. IEEE Trans. Intell. Transp. Syst **14**(4), 1743–1752 (2013)
5. Kavraki, L.E., Svestka, P., Latombe, J.-C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Rob. Automat **12**(4), 566–580 (1996)
6. Bohlin, R., Kavraki, L.E.: Path planning using lazy prm. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 521–528. IEEE (2000)
7. Nieuwenhuisen, D., Overmars, M.H.: Useful cycles in probabilistic roadmap graphs. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 446–452. IEEE (2004)
8. Marble, J.D., Bekris, K.E.: Computing spanners of asymptotically optimal probabilistic roadmaps. In: Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4292–4298. IEEE (2011)
9. Marble, J.D., Bekris, K.E.: Asymptotically Near-Optimal Planning With Probabilistic Roadmap Spanners. IEEE Trans. Rob **29**(2), 432–444 (2013)
10. Gayle, R., Sud, A., Andersen, E., Guy, S.J., Lin, M.C., Manocha, D.: Interactive Navigation of Heterogeneous Agents Using Adaptive Roadmaps. IEEE Trans. Visual. Comput. Graph **15**(1), 34–48 (2009)

11. Gayle, R., Sud, A., Lin, M.C., Manocha, D.: Reactive Deformation Roadmaps: Motion Planning of Multiple Robots in Dynamic Environments. In: Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3777–3783. IEEE, San Diego (2007)
12. Hilgert, J., Hirsch, K., Bertram, T., Hiller, M.: Emergency path planning for autonomous vehicles using elastic band theory. In: Proceedings of the 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, vol. 2, pp. 1390–1395. IEEE (2003)
13. Quinlan, S., Khatib, O.: Elastic bands: connecting path planning and control. In: Proceedings of the 1993 IEEE International Conference on Robotics and Automation, pp. 802–807. IEEE (1993)
14. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. Int. J. Robot. Res. **20**(5), 378–400 (2001)
15. Kuffner, J.J., LaValle, S.M.: RRT-connect: An efficient approach to single-query path planning. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 995–1001. IEEE (2000)
16. Lie, T.Y., Shie, Y.C.: An incremental approach to motion planning with roadmap management. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 4, pp. 3411–3416. IEEE (2002)
17. Morales, M., Rodriguez, S., Amato, N.: Improving the connectivity of PRM roadmaps. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 4427–4432. IEEE (2003)
18. Plaku, E., Bekris, K.E., Chen, B.Y., Ladd, A.M., Kavraki, E.E.: Sampling-Based Roadmap of Trees for Parallel Motion Planning. IEEE Trans. Rob **21**(4), 597–608 (2005)
19. Raveh, B., Enosh, A., Halperin, D.: A little more, a lot better: improving path quality by a path-merging algorithm. IEEE Trans. Rob **27**, 365–371 (2011)
20. Amato, N.M., Bayazit, O.B., Dale, L.K., Jones, C., Vallejo, D.: Obprm: An obstacle-based prm for 3d workspaces. In: Agarwal, P., Kavraki, L.E., Mason, M. (eds.) Robotics: The Algorithmic Perspective, pp. 155–168. A.K. Peters (1998)
21. Yeh, H.Y., Thomas, S., Eppstein, D., Amato, N.M.: UOBPRM: A uniformly distributed obstacle-based PRM. In: Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2655–2662. IEEE (2012)
22. Hsu, D., Jiang, T., Reif, J., Sun, Z.: The bridge test for sampling narrow passages with probabilistic roadmap planners. In: Proceedings of the 2003 IEEE International Conference on Robotics and Automation, vol. 3, pp. 4420–4426. IEEE (2003)
23. Boor, V., Overmars, M.H., van der Stappen, A.F.: The Gaussian sampling strategy for probabilistic roadmap planners. In: Proceedings of the 1999 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1018–1023. IEEE (1999)
24. Sun, Z., Hsu, D., Jiang, T., Kurniawati, H., Reif, J.H.: Narrow passage sampling for probabilistic roadmap planning. IEEE Trans. Rob **21**(6), 1105–1115 (2005)
25. Denny, J., Amato, N.M.: Toggle PRM: A Coordinated Mapping of C-free and C-obstacle in Arbitrary Dimension. In: Proceedings of the International Workshop on Algorithmic Foundations of Robotics, pp. 297–312. Springer, Boston (2012)
26. Wilmarth, S.A., Amato, N.M., Stiller, P.F.: MAPRM: A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1024–1031. Detroit (1999)
27. Holleman, C., Kavraki, L.E.: A framework for using the workspace medial axis in prm planners. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1408–1413. IEEE (2000)
28. Schmitzberger, E., Bouchet, J.L., Dufaut, M., Wolf, D., Husson, R.: Capture of homotopy classes with probabilistic road map. In: Proceedings of the 2002. IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, pp. 2317–2322. IEEE (2002)
29. Bhattacharya, S., Likhachev, M., Kumar, V.: Topological constraints in search-based robot path planning. Auton Rob **33**(3), 273–290 (2012)
30. Demyen, D., Buro, M.: Efficient triangulation-based pathfinding. In: 21st National conference on Artificial Intelligence, pp. 942–947. IEEE (2006)
31. Grigoriev, D., Slissenko, A.: Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane. In: Proceedings of the 1998 international symposium on Symbolic and algebraic computation, pp. 17–24. IEEE (1998)
32. Hsu, D., Sanchez-Ante, G., Sun, Z.: Hybrid PRM Sampling with a Cost-Sensitive Adaptive Strategy. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 3874–3880. IEEE (2005)
33. Shi, K., Denny, J., Amato, N.M.: Spark PRM: Using RRTs within PRMs to efficiently explore narrow passages. In: Proceedings of the 2014 IEEE International Conference on Robotics and Automation, pp. 4659–4666. IEEE (2014)
34. Siméon, T., Laumond, J.P., Nissoux, C.: Visibility-based probabilistic roadmaps for motion planning. Adv. Rob **14**(6), 477–493 (2000)
35. Jaillet, L., Cortes, J., Simeon, T.: Transition-based RRT for path planning in continuous cost spaces. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2145–2150. IEEE, France (2008)
36. Dobson, A., Bekris, K.E.: Sparse roadmap spanners for asymptotically near-optimal motion planning. Int. J. Rob. Res **33**, 18–47 (2014)
37. Littlefield, Z., Li, Y., Bekris, K.E.: Efficient sampling-based motion planning with asymptotic near-optimality guarantees for systems with dynamics. In: Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1779–1785. IEEE (2013)
38. Dobson, A., Krontiris, A., Bekris, K.E.: Sparse Roadmap Spanners. In: Algorithmic Foundations of Robotics X, Springer Tracts in Advanced Robotics, vol. 86, pp. 279–296. Springer (2013)
39. Ferguson, D., Stentz, A.: Anytime RRTs. In: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, pp. 5369–5375. IEEE, Beijing (2006)
40. Kalisiak, M., van de Panne, M.: RRT-blossom: RRT with a local flood-fill behavior, pp. 1237–1242. IEEE, Orlando (2006)

41. Strandberg, M.: Augmenting RRT-planners with local trees. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, vol. 4, pp. 3258–3262. IEEE (2004)
42. Urmson, C., Simmons, R.: Approaches for heuristically biasing RRT growth. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 1178–1183. IEEE, Las Vegas (2003)
43. Kala, R., Shukla, A., Tiwari, R.: Robotic Path Planning using Evolutionary Momentum based Exploration. J. Exp. Theor. Artif. Intell **23**(4), 469–495 (2011)
44. Bennewitz, M., Burgard, W., Thrun, S.: Optimizing schedules for prioritized path planning of multi-robot systems. In: Proceedings of the 2001 IEEE international conference on robotics and automation, pp. 271–276. IEEE (2001)
45. Bennewitz, M., Burgard, W., Thrun, S.: Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. Rob. Auton. Syst. **41**(2-3), 89–99 (2002)
46. Clark, C.M., Bretl, T., Rock, S.: Applying kinodynamic randomized motion planning with a dynamic priority system to multi-robot space systems. In: Proceedings of the 2002 IEEE Aerospace Conference Proceedings, vol. 7, pp. 3621–3631. IEEE (2002)

Rahul Kala received the B.Tech. and M.Tech. degrees in Information Technology from the Indian Institute of Information Technology and Management, Gwalior, India in 2010. He received his Ph.D. degree in Cybernetics from the University of Reading, UK in 2013. He is currently working as an Assistant Professor in the Indian Institute of Information Technology, Allahabad, India in the Robotics and Artificial Intelligence Laboratory. His areas of research are Multi-Robot Motion Planning, Autonomous Vehicles, Mission Planning for Mobile Robotics, Intelligent Transportation Systems, and related areas of Machine Learning, Soft Computing and Pattern Recognition. He is the author of three books published with CRC Press, Springer and IGI Global publishers respectively. He has also authored around 17 papers in SCI journals and over 80 papers in different journals and peer-reviewed conference proceedings. He is a recipient of the Best PhD dissertation award from the IEEE Intelligent Transportation Systems Society; a scholarship under the Commonwealth Scholarship and Fellowship Program from the UK Government; Lord of the Code Scholarship from RedHat and the Indian Institute of Technology Bombay; and the GATE scholarship from the Ministry of Human Resource Development, Government of India.