Robotica (2022), **40**, pp. 3104–3119
doi:10.1017/S026357472200008X

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Comparison of different sample-based motion planning methods in redundant robotic manipulators

Mehdi Shahabi[1], Hashem Ghariblu[1,*] and Manuel Beschi[2]

[1]Mechanical Engineering Department, University of Zanjan, Zanjan, Iran and [2]Italian National Research Council, Rome, Italy
*Corresponding author. E-mail: ghariblu@znu.ac.ir

Received: 8 August 2021; **Revised:** 4 December 2021; **Accepted:** 10 January 2022; **First published online:** 17 February 2022

Keywords: sample-based, motion planning, redundant, robotic manipulator

Abstract

The main objective of a motion planning algorithm is to find a collision-free path in the workspace of a robotic manipulator in a point-to-point motion. Among the various motion planning methods available, sample-based motion planning algorithms are easy to use, quick and powerful in redundant robotic systems applications. In this study, different sampling-based motion planning algorithms are employed to select the most appropriate method for efficient collision-free motion planning. As a case study, finding a collision-free robotic displacement for welding a main pipe with other intersecting pipes and joints is considered. The robotic manipulator employed in this study has seven degrees of freedom, where six degrees are related to the manipulator joints and one degree is related to its base linear movement suspended from ceiling. Five criteria, time, path length, path time, path smoothness and process time are used to evaluate the efficiency of different sample-based motion planning algorithms. Finally, a smaller set of more efficient algorithms are introduced based on the defined efficiency evaluation criteria.

1. Introduction

In many applications, like spot welding or multi-position welding in a large workspace, it is necessary that a robotic manipulator find a collision-free path to move from an earlier working point to the new working point [1–3]. The goal of motion planning in these cases is to find a collision-free path. On the other hand, these robotic applications have large workspaces, and the robot base must be movable, which causes motion planning to be more complex with respect to traditional robotic manipulators due to kinematic redundancy. Various methods have been introduced to plan robot paths between two points in the presence of obstacles such as Artificial Potential Field (APF) Method [4, 5], Search-based planning algorithms (SBP) [6], optimization methods and genetic algorithms (GA) [7–12]. They have some disadvantages, for example, getting stuck in local minima, no passage between closely spaced obstacles, oscillations in certain conditions for APF [13–16], deployment difficulty for SBP [17–20] and need a reduced robot model to decrease high computational cost for GA [7–12]. In comparison, other motion planning methods, called sample-based planning algorithms, are fast, easy for implementation and very powerful tool not only for planning in high-dimensional spaces, but also for system with complex dynamics.

A sampling-based motion planning can be stated as "the task of finding a collision free path in the state space of the robot from a distinct start state to a specific goal state, utilizing a path composed of configurations connected by collision free paths". As it will be discussed in detail in Section 2, Roadmap-based and Tree-based are two general concepts of sample-based motion planners. Researchers use different approach for sample-based motion planners and try to develop the algorithms. They do not necessarily have disadvantages, and the most of these algorithms are currently used in different research works. In this study, we will compare state-of-the-art sampling-based motion planning methods for the general case of point-to-point motion planning of robotic manipulators with static obstacles in their work space

© The Author(s), 2022. Published by Cambridge University Press.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

and their performances are compared with each other. Collision checking is a very important part of sampling-based planning [21]. Tsianos et al. [22] discussed the progress made in sampling-based algorithms in path planning. These algorithms guarantee the likelihood of finding an answer. Kaltsoukalas et al. [23] proposed an intelligent search algorithm to define the industrial robot path. Larsen et al. [24] presented a system that can plan paths for industrial robots and compared sampling-based methods with GAs. Kwon et al. [25] used sampling-based motion planning in a robotic system for autonomous grasping and manipulation. Alatortsev et al. [26] have shown sampling-based motion planning role in the approaches for optimization of the robotic task sequence.

In several robotic operations, it is needed for a robot to operate in different positions [27], for example, welding of different joints to the main pipe in the petrochemical and power plant industries or spot welding, assembling of different parts on the body of the vehicle industry. In these cases, after completing a task in one position, the robotic manipulator moves to the next position in order to do its new programmed task. In these applications due to the large workspace, the robot base is movable, and the robot has kinematic redundancy or excess degrees of freedom. Ghariblu and Javanmard find the maximum allowable load for two redundant robotic manipulators with kinematic redundancy in their common base [28]. Ghariblu and Shahabi [29] introduced a path planning algorithm for robotic systems with excess degrees of freedom for welding intersecting pipes. Beschi et al. [30] introduced an algorithm to generate optimal trajectories for redundant industrial robots in machining and additive manufacturing applications. Once motion planning algorithm finds the path, it gives its information to the control module. Our scope in this study is introducing motion planning algorithms that must be followed by sophisticated control methods to send appropriate commands to the actuators to follow the desired path [31, 32].

When a robot works at different points, two different kinds of movement for the robot are needed. First kind, which is called effective path, is the path of the end effector through a predefined trajectory to perform its operation in a place. The second kind, which is called supportive path, is the path of the robot between two task positions. The supportive path is not planned in a predefined path and should be planned between task points with motion planning algorithms, taking into account some path. Our main goal is finding an efficient supportive path between two task positions by comparing recently developed sample-based motion planning algorithms. The comparison between different motion planning algorithms is realized by different criteria, like shortest path, minimum time or consumed energy, maximum distance from peripheral obstacles, etc.

This work introduces a computational framework to analyze the efficiency of the most recent sampling-based methods for point-to-point motion planning for redundant robotic manipulators in large workspaces. Therefore, at first the state-of-the-art sampling-based motion planning methods for the general case of point-to-point motion planning are introduced. Then, these methods are implemented for a special case of displacing a welding torch from its initial position to the other task point in the neighborhood in the presence of static obstacles in the robot work space. In addition, an algorithm is proposed to choose the most appropriate method for manipulator motion in the neighborhood of the end points. The rest of the article is organized as follows. In Section 2, different sample-based path planning algorithms are introduced. The benchmarking is done in Section 3 to compare the performance of the various motion planning algorithms. Section 4 evaluates the results of motion planning algorithms to select a smaller set of suitable algorithms based on the defined criteria. Constrained collision-free path planning in the neighborhood of the end points is discussed in Section 5. Finally, Section 6 addresses conclusions and results obtained from this study.

2. Sample-based Path Planning Algorithms

As outlined in Introduction section, the sample-based path planning algorithms among the motion planning algorithms are easy to use, quick and powerful in redundant robotic systems because in a complex or high-dimensional system, it may not be easy to explicitly represent the free state space,

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

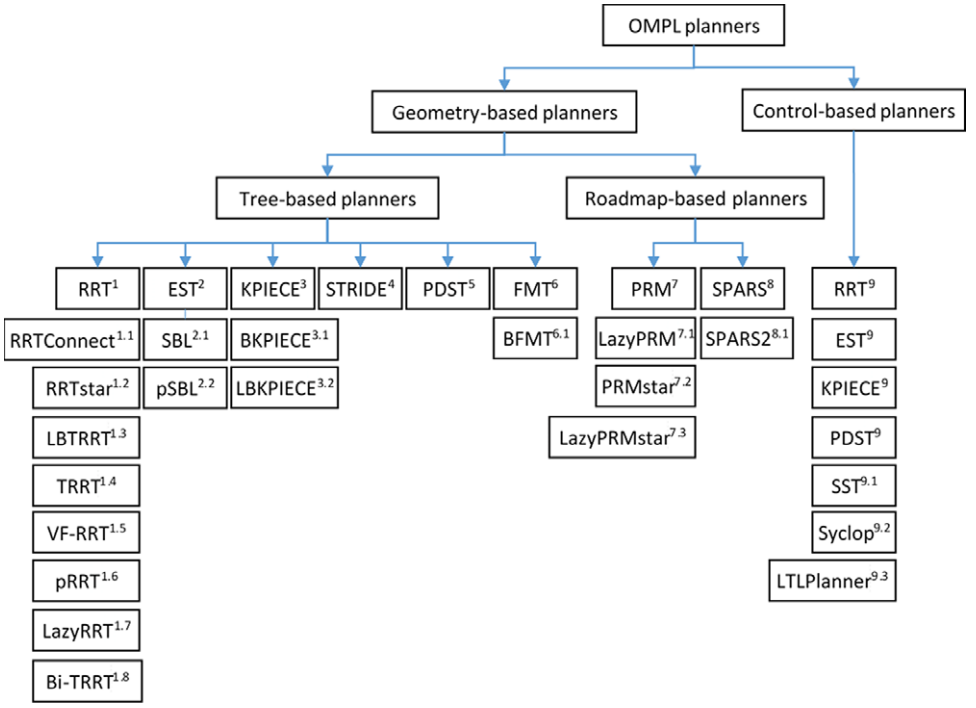


Figure 1. *Motion planners included in OMPL.*

but in sampling-based methods it is not necessary to create this space. Collision checking is used not only in the local planner when attempting to find collision-free paths between samples, but also during the sampling process itself. It is the job of the collision checker to accept a configuration of the robot and quickly determine whether this state is in collision. In the following sections, the supportive path planning algorithms and their corresponding efficiency are demonstrated.

Many sampling-based motion planners are collected in Open Motion Planning Library (OMPL) that is open source and can be accessed directly or via other software. Currently, it is available in the Moveit Software, which is state-of-the-art software for industrial robots based on the robot operating system (ROS) platform. The OMPL is a powerful collection of state-of-the-art sampling-based motion planning algorithms. Figure 1 shows the motion planners that exist in OMPL.

Based on whether workspace includes dynamic or differential constraints, geometry-based or control-based planners are used. A geometry-based planner just takes into account the geometry and kinematic constraints of the workspace and provide the output containing a series of point expressing the geometric configuration, that is, the coordinates in the joint space, while a control-based needs additional parameters, control space and state propagator, to apply control vector in order to control the propagation of the state.

Roadmap-based and Tree-based are two general concepts of geometry-based sample-based motion planners. As Fig. 2 shows, in Roadmap-based concept, the planner utilizes random sampling of the workspace to build a roadmap of the free workspace. Once the desired numbers of sampling points have been created, the roadmap itself can be constructed by connecting the random samples to form edges. Once the roadmap is complete, it can be used to answer motion planning queries by connecting the start and goal points to the roadmap using the local planner, and performing a graph search to find the shortest path in the roadmap. The probabilistic roadmap (PRM) is the most famous Roadmap-based motion planners [33].

In Tree-based concept as shown in Fig 3, the planner begins by rooting a tree at the starting configuration of the robot. The planner employs an expansion heuristic from which the sample is connected to

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

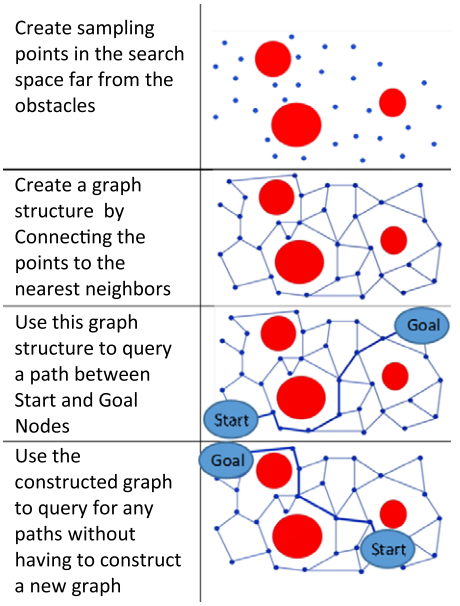


Figure 2. *PRM algorithm.*

the tree along a collision-free path. The Rapidly exploring Random Tree known as RRT [34] is the basic Tree-based motion planner. As it is shown in Fig (4), RRTstar [35], is one of the most famous variant of RRT, that uses optimization steps to add nodes to the tree. Two main optimization objectives are minimizing the Path Length and maximizing the clearance to obstacles. More optimization objective has been created by combination of these two objective such as Balanced Optimization, which optimizes a trade-off between path length and maximum clearance via weighting factor, and Maximize minimum clearance, that is the cost of a given path is the only function of the closest distance between the path and an obstacle.

In this study, a problem with static constraints is considered, so geometry-based planners are used. Before describing the problem, a brief description of the used planners is reviewed.

1- RRT (Rapidly-exploring Random Tree) [32]: as described in Fig. 3 it is the basic tree-based technique. There are some variants of RRT, since these variant are repeated for other planners we explain them as follows:

1.1- RRTConnect [36]: The bi-directional version of the RRT, where two trees are grown simultaneously from start and goal nodes until they meet each other.

1.2- RRTstar [35]: The RRT planner with an optimizing step, where a new sample node is connected to the tree so that the state space is more locally refined.

1.3- LBTRRT (Lower Bound Tree-RRT) [37]: It is an asymptotically near-optimal version of RRT that uses an auxiliary graph called lower bound graph.

1.4- TRRT (Transition-based RRT) [38]: The proposed method combines the exploratory strength of RRTs, with the efficiency of stochastic optimization methods. It tries to find short, low-cost paths by performing transition tests to add new states to the tree.

1.5- VF-RRT (Vector Field RRT) [39]: tries to minimize the so-called upstream cost of a path. The upstream cost is defined by an integral over a user-defined vector field.

1.6- pRRT (Parallel RRT) [40]: Many different parallelization schemes have been proposed so that several threads simultaneously add states to the same tree. Once a solution is found, all threads terminate.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

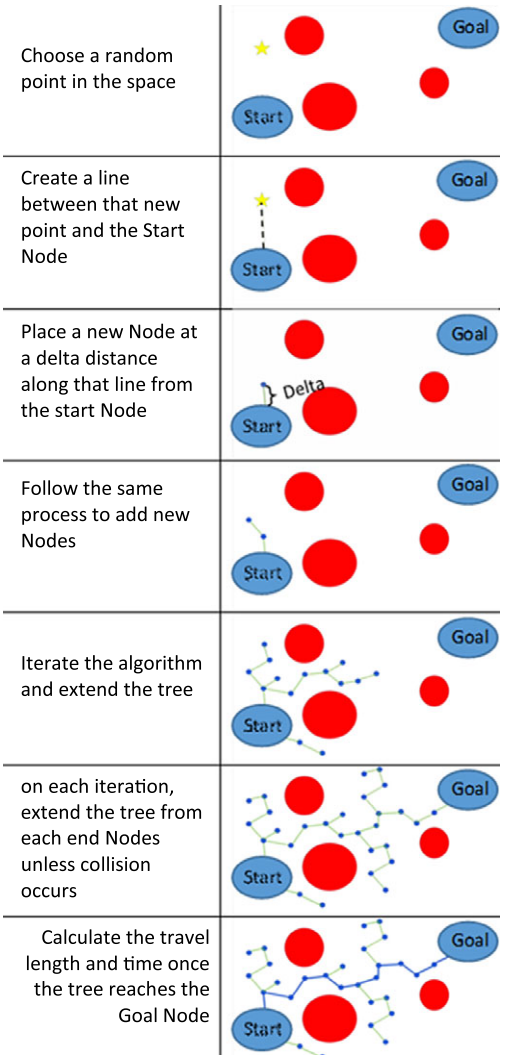


Figure 3. *RRT algorithm.*

1.7- LazyRRT (Lazy RRT): It checks the validity of a state "lazily" so that accepts invalid configuration states to construct a tree. After the graph search is performed, the invalid parts of the candidate solution are altered to make the path collision-free.

1.8- Bi-TRRT [41]: The Optimized bi-directional version of TRRT.

2- EST (Expansive Space Trees) [42]: This planner was published around the same time as RRT. It tries to determine the direction of the tree by checking the density of nodes in the configuration space. The tree will expand towards the less explored space. Like RRT, there are bi-directional, lazy collision checking, projection-based and parallel variants of EST.

2.1- BiEST [42]: Bi-directional variant of EST

2.2- ProjEST [42]: Projection variant of EST

2.3- SBL [43]: It is single-query bi-directional lazy collision checking

2.4- Psbl [43]: It is Parallel Single-query Bi-directional Lazy collision checking variant

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

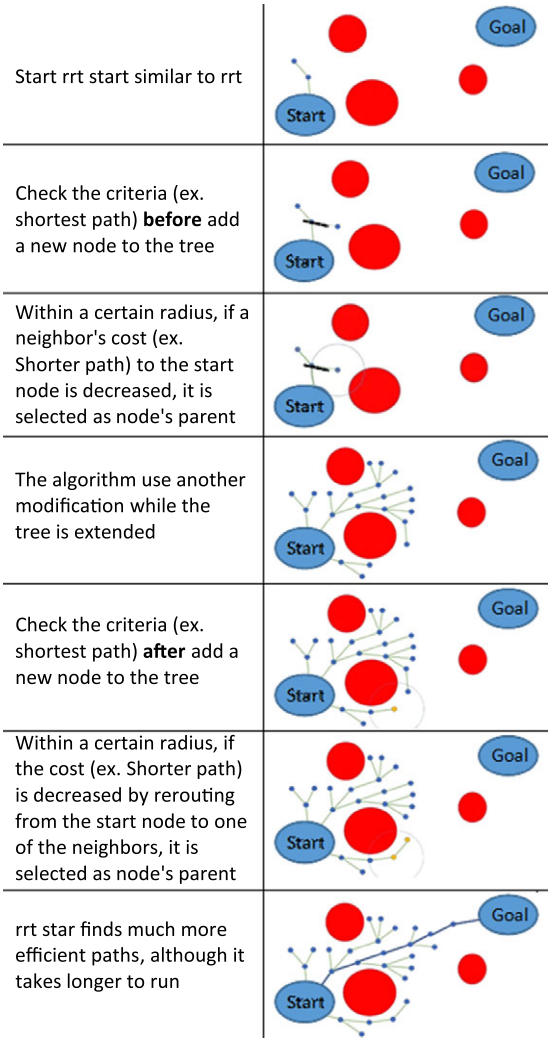


Figure 4. *RRTstar algorithm.*

3- KPIECE (Kinodynamic motion Planning by Interior-Exterior Cell Exploration) [44]: This planner uses layers of discretization to guide the exploration of the (continuous) state space. There are two variants of KPIECE, bi-directional and lazy collision checking.

3.1- BKPIECE: Bi-directional variant of KPIECE
3.2- LBKPIECE: Lazy Bi-directional variant of KPIECE

4- STRIDE (Search Tree with Resolution Independent Density Estimation) [45]: This planner was inspired by EST. Instead of using a projection, STRIDE uses a Geometric Near-neighbor Access Tree to estimate sampling density directly in the state space. In comparison to the EST, STRIDE is useful for high-dimensional systems where the free space cannot easily be captured with a low-dimensional (linear) projection.

5- PDST (Path-Directed Subdivision Tree) [46]: this planner represents samples as path segments instead of configuration states. It uses non-uniform subdivisions called cell to explore the state space. The density of motions per cell is used to guide expansion of the tree. PDST entirely

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

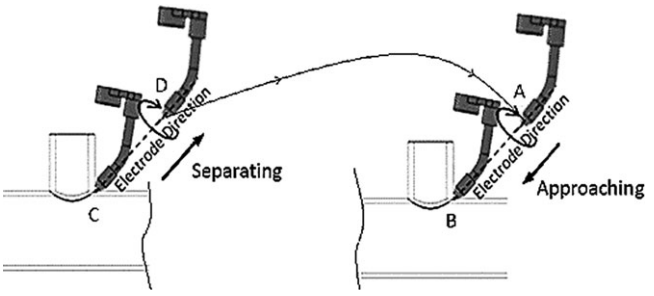


Figure 5. *Supportive path between two task position, Phase 1) Separating from first task position, Phase 2) Moving through the path between the two task positions, and Phase 3) Approaching to the second task position.*

removes the dependency on a distance measure, which is useful in cases where a good distance metric is hard to define.

6- FMT (Fast Marching Tree) [47]: this planner is an asymptotically optimal planner, which marches a tree forward in the cost-to-come space on a specified amount of samples. Unlike all other planners, the numbers of valid samples need to be chosen beforehand.

6.1- BFMT (bi-directional Fast Marching Tree) [48]: The bi-directional variant of FMT.

7- PRM (Probabilistic Road Map) [30]: as described in Fig. 2, it is the basic Probabilistic Road Map technique. There are lazy collision checking, star and lazy plus star variants of PRM.

7.1- LazyPRM [49]: Lazy collision checking variant of PRM

7.2- PRMstar [35]: star variant of PRM

7.3- LazyPRMstar [35]: It is PRMstar with lazy collision checking.

8- SPARS [50]: This planner provides asymptotic near-optimality and includes a meaningful stopping criterion. Its convergence rate tends to be much higher than PRMstar.

8.1- SPARS2 [51]: It is a variant of SPARS, that works through similar mechanics, but uses a different approach to identifying interfaces and computing shortest paths.

3. Path Segmentation in the Motion Planning

As it is seen in Fig. 5, in the large workspaces motion of an end effector between two task points can be divided into three phases. At the first phase, the end effector moves away from the first task point (C to D in Fig. 5), then in the second phase it moves toward the next task positions (D to A), and finally in the phase 3, it approaches to the next task point (A to B). Therefore, an auxiliary point, marked as A and D in Fig. 5, is defined near the starting/ending point of task points (C and B). In phase 1 and phase 3, the robot moves with path constraint. This path constraint keeps the orientation of the end effector. Robot movements in phase 1 and phase 3 are slow, while the robot moves with maximum speed in the phase 2.

Path planning considering path constraint (C-D and A-B in Fig. 5) has a high computing cost, because for each state (sample) the inverse kinematic has to be solved. Therefore, in this study, the problem is divided into some steps, which are shown in Fig. 6.

In the first step, for a specific path from A to D without regarding the auxiliary points, the benchmark between different sample-based motion planning algorithms is done to compare the performance of the various motion planning algorithms. In the second step, the path planned by the selected algorithms of the prospective step is examined to ensure satisfying the kinematic constraints and the smoothness of the path. In the third step, the constrained path planning between the auxiliary points and the starting points (C-D and B-A paths) is demonstrated.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

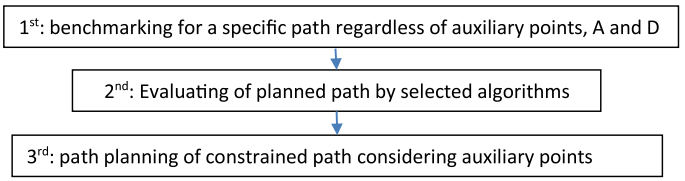


Figure 6. *The motion planning is divided into some steps.*

4. Benchmarking

In this section for benchmarking, the same problem is used to compare different motion planning algorithms which are described in Section 2. First, as it is depicted in Fig. 6 and Table I, our problem is defined to move from start state to goal state. After the stop condition of runs is introduced, the number of missing runs is counted up due to this stop condition. Finally, different sample-based algorithms are compared using the results of runs.

To implement a benchmarking, some parameters have to be defined in Moveit. The main parameters are Planning scene, Group, Start sates, Queries, Workspace (Optional), Goal constraints (Optional), Path constraints (Optional) and Planners which consist of Plugin and Planner-Name.

Where Group determines the group of robot joints participating in motion planning. Queries represent the joints values in the goal state. Start states represent the joints values in the start state. Workspace determines the space that planners search in. plugin support motion planner plugin like OMPL. Planner-Name is the name of planner, in case of OMPL one or some of the planner name of Fig. 1. The Planning scene transforms information such as robot_state, which determines the name, angle (position), velocity and effort of joints, robot_model_name, which includes the geometry of link and joints of robot, fixed_frame_transforms, which determines the reference frame, and allowed_collision_matrix, which determines the pair links that their collisions are ignored.

As it is shown in Fig. 7, the robotic manipulator "Cloos Romat 320" employed in this case study has seven degrees of freedom (DOF), joint number zero belongs to the base of the manipulator linear movement suspended from ceiling, and another six degrees of freedom belongs to the manipulator joints. This robotic manipulator is redundant for various tasks like arc or spot welding and cutting operations, because these operations only need 5 or 6 DOF. For the benchmarking of case study, Fig. 7 shows the start state and goal state, which are indicated by the robot joint angles. Table I shows the robot joint angle limits. To avoid repetition and extension of this paper, the kinematic analysis of this manipulator we refer it to the authors previously published paper [1].

4.1. Case study

For analysis and comparing performance of the different sampling-based motion planning methods between start and goal states, a case study is considered. In this case study, Cloos Romat 320 manipulator moves from start state: $[X_s, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]_{start} = [0.111cm, 0°, –127.2°, –12°, 38.4°, 0°, 140.9°]$ to the goal state: $[X_s, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]_{goal} = [-0.103cm, 0°, 0.6°, –160.4°, –73.9°, 0°, 73.9°]$. The first term Xs indicates the robot base position in centimeter, while other six terms, that is, [1, 2, 3, 4, 5, 6] indicate manipulator joints angles in radians. For goal state, a general default tolerance of 1e-5 length unit in position and 1e-5 radians in orientation is specified in Moveit. These tolerances are sufficient for many robotic applications like welding. There is no path constraint in this case. The request for benchmarking is done using the motion planning request message. This message transfers information such as workspace dimensions, start state, goal constraints and allowed planning time (10 s).

The benchmarking collects various statistics for different types of planners. In this case, 23 planners are used and 50 runs per planner are executed with a 10 s time limit for each run. Memory limit for each

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

Table I. *Robot joint angles limits of industrial welding robot CLOOS ROMAT 320 [1].*

Joint no.	Joint angles limits
0	-50 cm < d0 < $+50$ cm
1	$-170^{o} > \theta 1 < +170^{o}$
2	$-180^{o} > \theta 2 < +35^{o}$
3	$-210^{o} > \theta 3 < +60^{o}$
4	$-179^{o} > \theta 4 < +179^{o}$
5	$-135^{o} > \theta 5 < +135^{o}$
6	$-300^{o} > \theta 6 < +300^{o}$

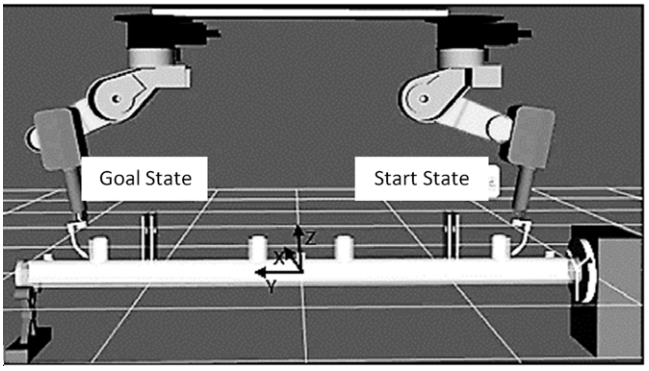


Figure 7. *Start state and goal state simulated in Moveit.ROS.*

run was 100 KB. Totally, 86.25 min spent to collect the data of this benchmarking using an Intel core i7 − 6500 U and 8 GB memory computer.

After introducing the stop condition, the missing runs are counted up due to this stop condition and results are shown in Table II. Also Fig. 8 compares the different performance criteria between various sample-based algorithms.

If either planner could not find the solution in 10 s or its memory exceeds 100KB, the related run will be missed. Table II shows the number of missing runs. Sparstwo and TRRT have many missed runs. Hence, these two algorithms were omitted for the next runs.

To show the statistical distribution, the box plot is used that is particularly useful for comparing distributions between several groups or sets of data. Figure 8 shows the results of benchmarking.

Path length in centimeter, path time in seconds, path smoothness and process time in seconds are five factors, which demonstrate the state of each planner. The less smoothness value means the smoother movement. The log scale for Y-axis is used to have more visible data. Analyzing the obtained results, the following conclusions are made:

- Among roadmap-based planners, the path specifications, that is, path time, path length and path smoothness of LazyPRM and LazyPRMstar are better than what was found by PRM, PRMstar and Spars. However, the process time of PRM, PRMstar and Spars are better than LazyPRM and LazyPRMstar.
- A trade-off between roadmap-based and tree-based planners demonstrates that although the roadmap-based planner, that is, PRM, LazyPRM, PRMstar, LazyPRMstar and Spars found better solutions regarding the path length and path time, their computational cost is much higher than tree-based planners. Therefore, in our application, tree-based planners are preferred to the roadmap-based planners.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

Table II. *The number of missing runs of different planner in 50 runs.*

Item no.	Planner name	Missing runs	Total runs
1	BKPIECE	0	50
2	BIEST	0	50
3	BITRRT	0	50
4	EST	0	50
5	FMT	0	50
6	KPIECE	0	50
7	LBKPIECE	0	50
8	LazyPRM	0	50
9	LazyPRMStar	0	50
10	PDST	16	50
11	PRM	0	50
12	PRMStar	0	50
13	ProjEST	0	50
14	RRT	0	50
15	RRTConnect	0	50
16	RRTStar	15	50
17	SBL	0	50
18	SPARS	0	50
19	SPARSTwo	48	50
20	STRIDE	0	50
21	TRRT	44	50

- A trade-off between tree-based planners demonstrates that BiTRRT could find the best solution. Because its path specifications are more and less similar to RRT, RRtstar, PDST, and FMT, but its computational cost is much less than those are. However, The RRTstar could find the best path specifications.

As a conclusion, BiTRRT is selected as the most appropriate planner. BiTRRT is the bi-directional version of TRRT, which uses a stochastic optimization method and tries to add new states to the tree in order to find short, low-cost paths by performing transition tests. However, RRTstar is the better choice if only the shortest path is desired. RRTstar is an RRT planner with an optimizing step, where a new sample node is connected to the tree so that the state space is more locally refined.

5. Evaluation of the Path Planned by the Selected Motion Planning Algorithms

In this section, we will look more closely at the paths planned by selected motion planning algorithms. Each BiTRRT and RRTstar algorithms is executed five times with the same start and goal states. The best execution is chosen and its simulation in the Cartesian space is shown in Fig. 9. As it is shown in a circle in Fig (9), the tool tip paths are different for BiTRRT and RRTstar algorithms in the Cartesian space. Here, the robot is requested to move from the auxiliary point D with the coordinates $[0, -0.42, 0.16]$ and Euler angles $\left[45^\circ, 0^\circ, 0^\circ\right]$ to auxiliary point A of another welding position with the coordinates $[0.16, 0.42, 0]$ and the Euler angles $\left[-45^\circ, 0^\circ, 0^\circ\right]$.

The number of path samples and length of traveled path planned by the selected motion planning algorithms BiTRRT and RRTstar are stated in Table III. Because there are some obstacle like two branch pipe and also main pipe between start and goal states, the BiTRRT and RRTstar algorithms use different strategy and therefore the number of path samples is significantly different. In this case, RRTStar succeed to reach goal state with fewer path samples. It means that the computational cost is lower and it is faster

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

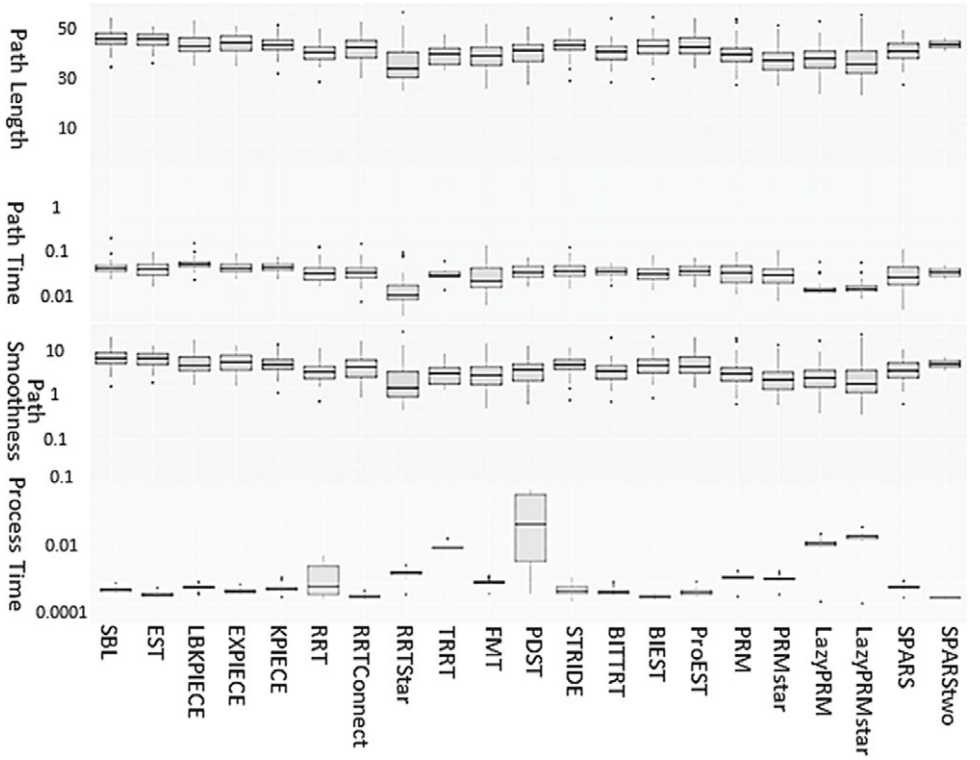


Figure 8. *The benchmarking results of different type of sample-based motion planning algorithms.*

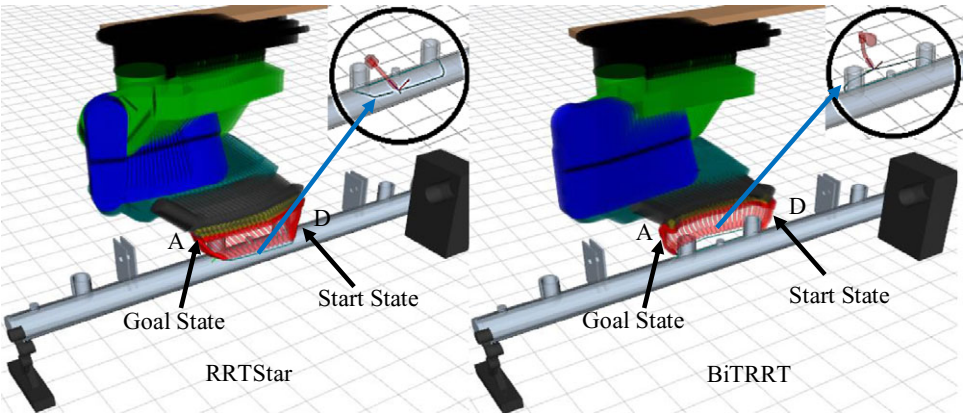


Figure 9. *Simulation of the path planned by the selected path planning algorithms with the same start and goal states in Moveit.ROS.*

than BiTRRT. Although the length of traveled path by RRTStar is 5.7 cm (108.3–102.6) longer than BiTRRT, but as Fig. 9 shows it is smoother than BiTRRT's one.

Figure 10 shows the joint angles along the path planned by the selected motion planning algorithms. In this figure, *q_0* represents the linear motion of the robot base in cm, q_1 to q_6 are the joint angles of the robot joints. According to Table I, all joint angles are within the range of the production path. Also, as the figure shows, the path planned by both the BiTRRT and the RRTstar algorithms in the joint space are smooth.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

Table III. *Path specification planned by selected algorithms BiTRRT and RRTstar.*

Motion planning algorithm	Length of traveled path (cm)	Number of path samples	Path time (second)
BiTRRT	102.6	61	40.741
RRTstar	108.3	36	22.302

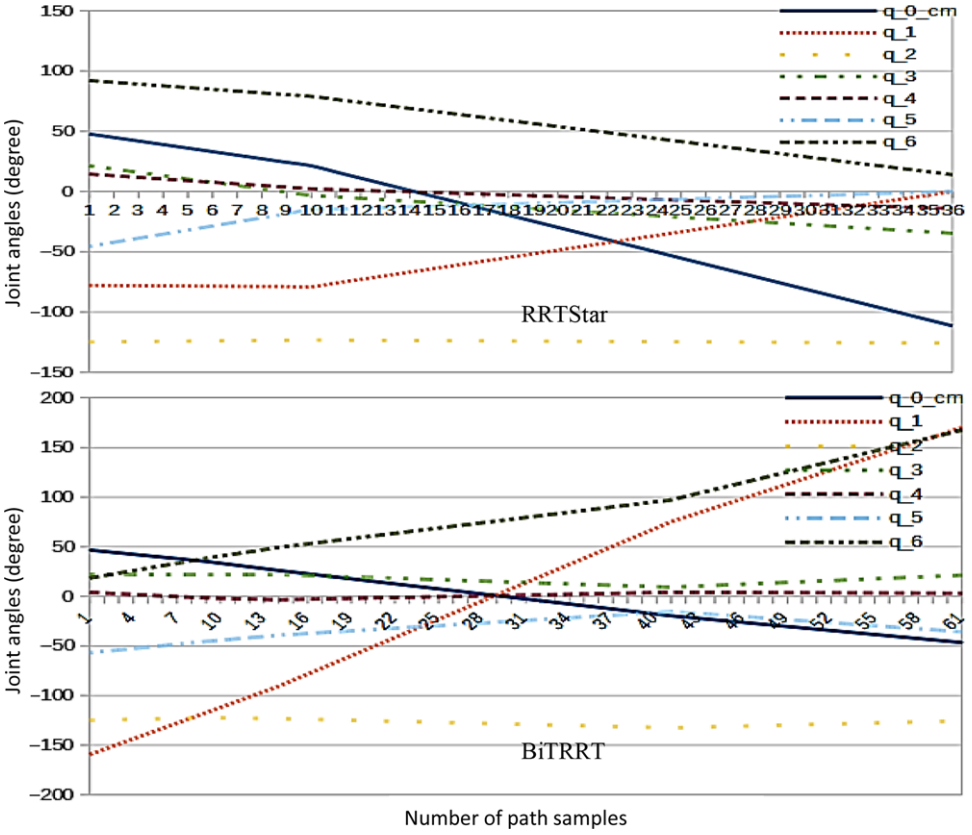


Figure 10. *Joints angles along the path planned.*

6. Constrained Path Planning in the Neighborhood of the Welding Positions

In this section, the end effector keeps its orientation while it separates or approaches towards the task positions. The robot travels the path with a low speed to ensure collision avoidance. In this section, the ability of selected motion planning algorithms, BiTRRT and RRTstar, is examined in order to plan these constrained paths. Like previous section, each motion planning algorithm is executed five times and the best execution is considered.

Figure 11 illustrates the simulation of robot motions in the path planned in the Cartesian space twice for each selected motion planning algorithm, BiTRRT and RRTstar. Here, the end effector of robot was requested to travel from the point A with the coordinates [0, −0.42, 0.16] and Euler angles $[45^\circ, 0^\circ, 0^\circ]$ to the starting point of task position B with the coordinates [0, −0.37, 0.11] and Euler angles $[45^\circ, 0^\circ, 0^\circ]$, while keeps the end effector orientation during path traveling.

The number of path samples and length of traveled path planned by the selected motion planning algorithms BiTRRT and RRTstar are stated in Table IV. Because there are no obstacles between start

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

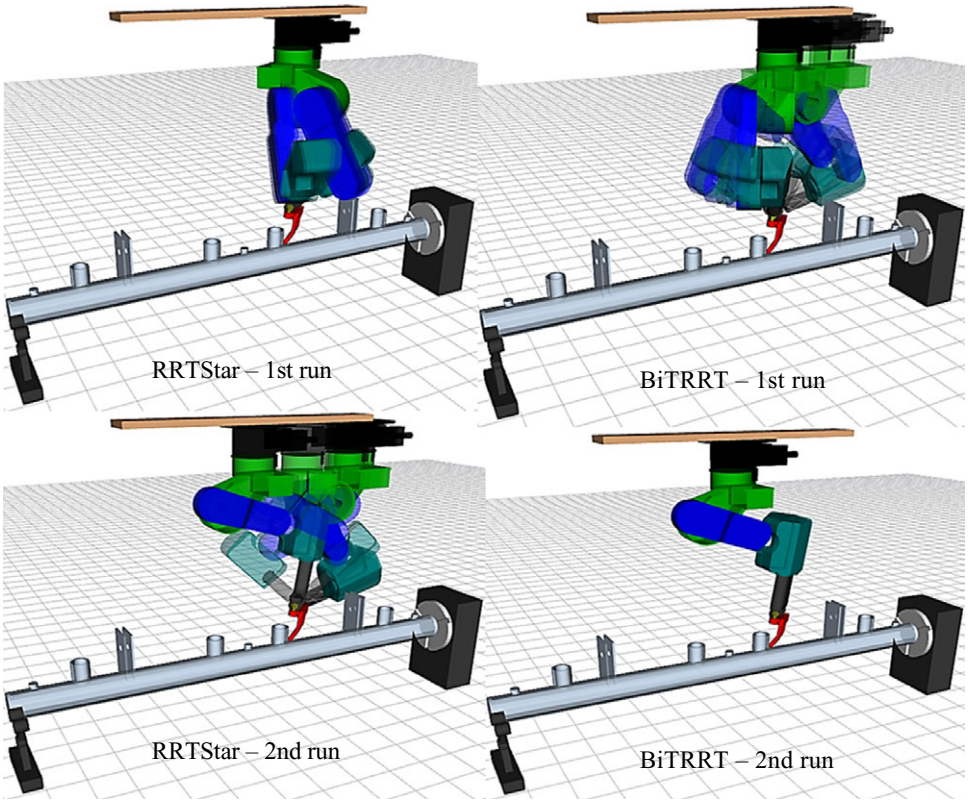


Figure 11. *Constrain path planned simulation, keeping the end effector orientation during path, twice for BiTRRT and RRTstar.*

and goal states, the number of path samples does not differ significantly. They use minimum number of samples, assumed 10, to find the path. The Euclidean distance between start state [0, −0.42, 0.16] and goal state [0, −0.37, 0.11] in Cartesian space is 0.0707. It is the minimum length of travel. The reported length in BiTRRT is less than this amount. It is probably due to the accumulation of rounding errors of distance between samples calculated by the BiTRRT algorithm. However, the difference in length of traveled path indicates that robot end effector do not move in the straight line. Another point is that the path time is not directly related to the length of travel. The 1st run of BiTRRT is longer but faster than the 2nd run of BiTRRT, because the path time depends on the joint angle movement in the joint space. Less joint angle changes cause less path time.

Figure 12 shows the joint angles along the path planned by the selected motion planning algorithms. In this figure, *q_0* represents the linear motion of the robot base in cm, q_1 to q_6 are the joint angles of the robot joints. According to Table I, all joint angles are within the range of the production path.

Among all four planned paths, the BiTRRT_02 has the best results because the joint angles have minimal motion and are smooth. Figure 12 shows the reason for difference between the planned paths, which is the configuration, or joint angles, at the starting points of the path. Therefore, by choosing the right joint angles at the starting points of the constrained path, an acceptable and smooth path can be planned.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

Table IV. *Path specification planned by selected algorithms BiTRRT and RRTstar.*

Motion planning algorithm	Length of traveled path (cm)	Number of path samples	Path time (second)
BiTRRT – 1st run	6.14	10	10.236
BiTRRT – 2nd run	6.07	10	18.478
RRTstar – 1st run	7.06	10	13.003
RRTstar – 2nd run	7.20	10	18.909

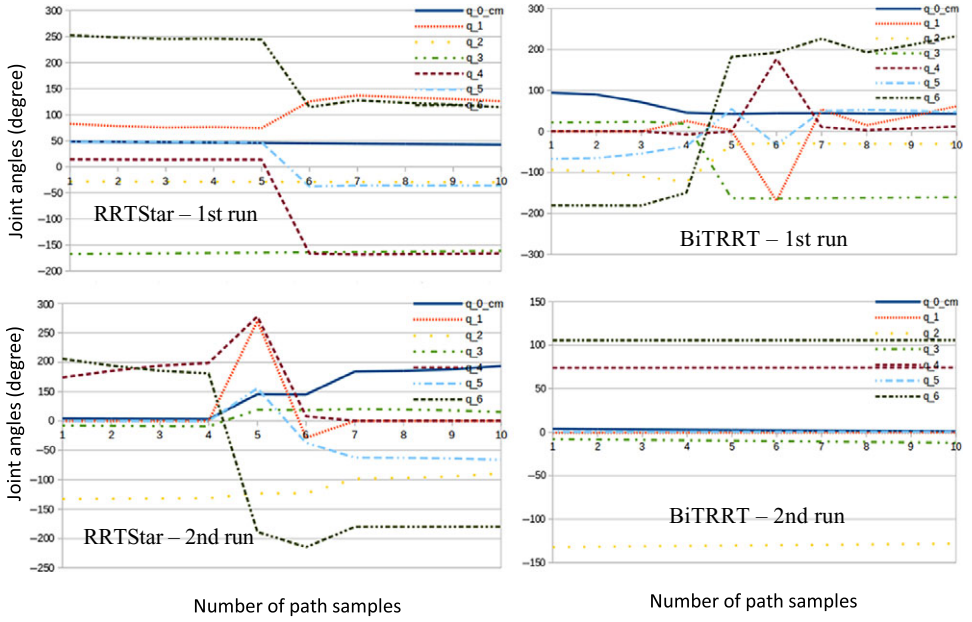


Figure 12. *Joints angles along the constriant path.*

7. Conclusion

In this study, different sample-based path planning for point-to-point robotic manipulator motion to travel between two task points was considered. These motion planning methods can be employed for large workspaces that need to use a redundant robotic manipulator with linear movement in its base. In comparison with other related researches, this work has taken in to account most important state-of-the-art sampling-based motion planning methods and compares their performances. Different criteria, like running time, number of successful paths, path time, path length and path smoothness were employed for performance analysis and comparison. Hence, 23 kinds of planners are used and 50 runs per planner are executed with 10 s time limit for each run, and 100 KB memory limit for each run using an Intel core i7 – 6500 U and 8 GB memory computer. The following conclusions for this analysis were obtained.

- Results demonstrate that roadmap-based planners found better solutions regarding the path specification, that is, path length and path time.
- On the other hand, computational cost of Tree-based planners are much lower than roadmap-based planners, So they are more appropriate where fast path planning analysis is required.
- Between Tree-based planners, Bi-directional Transition-based RRT (BiTRRT) and RRTstar have superior performances in comparison with other RRT methods. Therefore, BiTRRT and RRTstar are selected among 23 path planners for obstacle-free path planning.

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

- Finally, it is seen that BiTRRT has better results with minimal joint angles, motion and smoothness of path. In addition, it is seen that for the same sample points (nodes), the length of the traveled path of BiTRRT is shorter than RRTstar.

References

1. M. Shahabi and H. Ghariblu, "Optimal joint motion for complicated welding geometry by a redundant robotic system," *Eng. Optim.* **52**(5), 875–895 (2020).
2. M. Shahabi, H. Ghariblu and M. Beschi, "Obstacle avoidance of redundant robotic manipulators using safety ring concept," *Int. J. Comput. Integr. Manuf.* **32**(7), 695–704 (2019).
3. M. Shahabi, H. Ghariblu, M. Beschi and N. Pedrocchi,"Path planning methodology for multi-layer welding of intersecting pipes considering collision avoidance," *Robotica* **39**(6), 945–958 (2021).
4. O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Proceedings. 1985 IEEE International Conference on Robotics and Automation* (1985) IEEE.
5. Y. Mingxin, W. Sun'an, W. Canyang and L. Kunpeng, "Hybrid ant colony and immune network algorithm based on improved APF for optimal motion planning," *Robotica* **28**(6), 833–846 (2010).
6. M. Likhachev, *Search-based planning library(SBPL)* (2016).
7. M. M. irzaei Teshnizi, A. Kosari, S. Goliaei and S. Shakhesi, "Centralized Path Planning for Multi-aircraft in the Presence of Static and Moving Obstacles," *Int. J. Eng.* **33**(5), 923–933 (2020).
8. A. Elshamli, H. A. Abdullah and S. Areibi, "Genetic algorithm for dynamic path planning," *Electrical and Computer Engineering, 2004. Canadian Conference on* (2004) IEEE.
9. S. Sadegh Moghadasi and N. Faraji, "An efficient target tracking algorithm based on particle filter and genetic algorithm," *Int. J. Eng.* **32**(7), 915–923 (2019).
10. F. J. M. Abu-Dakka, Trajectory planning for industrial robot using genetic algorithms (2011).
11. Z. S. Abo-Hammour, O. M. Alsmadi, S. I. Bataineh, M. A. Al-Omari and N. Affach, "Continuous genetic algorithms for collision-free cartesian path planning of robot manipulators," *Int. J. Adv. Robot. Syst.* **8**(6), 74 (2011).
12. A. C. Nearchou and N. A. Aspragathos, "A genetic path planning algorithm for redundant articulated robots," *Robotica* **15**(02), 213–224 (1997).
13. Y. Koren, and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on* (1991) IEEE.
14. C. I. Connolly, J. B. Burns and R. Weiss, "Path planning using Laplace's equation," *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on* (1990) IEEE.
15. S. S. Ge, and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Trans. Robot. Autom.* **16**(5), 615–620 (2000).
16. L. Keselman, *Motion Planning for Redundant Manipulators and Other High Degree-of-Freedom Systems* (Georgia Institute of Technology, Georgia, USA, 2014).
17. M. Likhachev and D. Ferguson, "Planning long dynamically feasible maneuvers for autonomous vehicles," *Int. J. Robot. Res.* **28**(8), 933–945 (2009).
18. M. Pivtoraiko, D. Mellinger, and V. Kumar, "Incremental micro-UAV motion replanning for exploring unknown environments," *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (2013) IEEE.
19. M. Pivtoraiko and A. Kelly, "Kinodynamic motion planning with state lattice motion primitives," *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on* (2011) IEEE.
20. M. Pivtoraiko, R. A. Knepper and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *J. Field Robot.* **26**(3), 308–333 (2009).
21. G. Kang, B. K. Yong, H. L. Young, S. O. Hyun, S. Y. Won and R. C. Hyouk, "Sampling-based motion planning of manipulator with goal-oriented sampling," *Intell. Serv. Robot.* **12**(3), 265–273 (2019).
22. K. I. Tsianos, I. A. Sucan and L. E. Kavraki, "Sampling-based robot motion planning: Towards realistic applications," *Comput. Sci. Rev.* **1**(1), 2–11 (2007).
23. K. Kaltsoukalas, S. Makris and G. Chryssolouris, "On generating the motion of industrial robot manipulators," *Robot. Comput. Integr. Manuf.* **32**, 65–71 (2015).
24. L. Larsen, J. Kim, M. Kupke and A. Schuster, "Automatic path planning of industrial robots comparing sampling-based and computational intelligence methods," *Procedia Manuf.* **11**, 241–248 (2017).
25. M. Kwon, D. Zhou, S. Liu and H. Zhang, "A Robotic System for Autonomous Grasping and Manipulation," **In:** *First Robotic Grasping and Manipulation Challenge* (Yu Sun and Joe Falco, eds.) (Springer, Cham, 2016) pp. 816.
26. S. Alatartsev, S. Stellmacher and F. Ortmeier, "Robotic task sequencing problem: A survey," *J. Intell. Robot. Syst.* **80**(2), 279–298 (2015).
27. T. Zhu, X. Tang, C. Chen, F. Peng, R. Yan, L. Zhou, Z. Li and J. Wu, "High precision and efficiency robotic milling of complex parts: challenges, approaches and trends" *Chinese J. Aeronaut.* **35**(2), 22–46 (2022).
28. A. Javanmard and H. Ghariblu, "The effect of base position on maximum allowable load of dual arm robots," *Int. J. Eng.* **28**(2), 284–294 (2015).
29. H. Ghariblu and M. Shahabi, "Path planning of complex pipe joints welding with redundant robotic systems," *Robotica* **37**(6), 1020–1032 (2019).

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press

30. M. Beschi, S. Mutti, G. Nicola, M. Faroni, P. Magnoni, E. Villagrossi and N. Pedrocchi, "Optimal robot motion planning of redundant robots in machining and additive manufacturing applications," *Electronics* **8**(12), 1437 (2019).
31. W. Deng, J. Yao, Y. Wang, X. Yang and J. Chen, "Output feedback backstepping control of hydraulic actuators with valve dynamics compensation," *Mech. Syst. Signal Process.* **158** (2021).
32. W. Deng and Yao, J. "Asymptotic tracking control of mechanical servosystems with mismatched uncertainties," *IEEE/ASME Trans. Mechatron.* **26**(4), 2204–2214 (2020).
33. L. Kavraki, P. Svestka and M. H. Overmars, *Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. IEEE Transactions on Robotics and Automation*, **12**(4), 566–580 (1996).
34. S. M. LaValle, Rapidly-exploring random trees: A new tool for path planning. The annual research report (1998): n. pag.
35. S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.* **30**(7), 846–894 (2011).
36. J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," *Robotics and Automation, 2000. Proceedings. ICRA′00. IEEE International Conference on* (2000) IEEE.
37. O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality motion planning," *IEEE Trans. Robot.* **32**(3), 473–483 (2016).
38. L. Jaillet, J. Cortés and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robot.* **26**(4), 635–646 (2010).
39. I. Ko, B. Kim and F. C. Park, "VF-RRT: Introducing optimization into randomized motion planning," *Control Conference (ASCC), 2013 9th Asian* (2013) IEEE.
40. J. Bialkowski, S. Karaman and E. Frazzoli, "Massively parallelizing the RRT and the RRT," *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on* (2011) IEEE.
41. D. Devaurs, T. Siméon and J. Cortés, "Enhancing the transition-based RRT to deal with complex cost spaces," *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (2013) IEEE.
42. D. Hsu, J.-C. Latombe and R. Motwani, "Path planning in expansive configuration spaces," *Proceedings of International Conference on Robotics and Automation* (1997) IEEE.
43. G. Sánchez and J.-C. Latombe, "*A single-query bi-directional probabilistic roadmap planner with lazy collision checking*," **In:** *Robotics Research* (Raymond Austin Jarvis and Alexander Zelinsky, eds.) (Springer, Berlin, Heidelberg, 2003) pp. 403–417.
44. I. A. Şucan and L. E. Kavraki, "Kinodynamic motion planning by interior-exterior cell exploration," **In:** *Algorithmic Foundation of Robotics VIII,* Springer Tracts in Advanced Robotics (Gregory S. Chirikjian, Howie Choset, Marco Morales and Todd Murphey, eds.) (Springer, Berlin, Heidelberg) pp. 449–464.
45. B. Gipson, M. Moll and L. E. Kavraki, "Resolution independent density estimation for motion planning in high-dimensional spaces," *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (2013) IEEE.
46. A. M. Ladd and L. E. Kavraki, "Motion Planning in the Presence of Drift, Underactuation and Discrete System Changes," **In:** *Robotics: Science and Systems* (2005).
47. L. Janson, E. Schmerling, A. Clark and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *Int. J. Robot. Res.* **34**(7), 883–921 (2015).
48. J. A. Starek, V. Gomez, E. Schmerling, L. Janson, L. Moreno and M. Pavone. "An asymptotically-optimal sampling-based algorithm for bi-directional motion planning," *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on* (2015) IEEE.
49. R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," *Robotics and Automation, 2000. Proceedings. ICRA′00. IEEE International Conference on* (2000) IEEE.
50. A. Dobson and K. E. Bekris, "Sparse roadmap spanners for asymptotically near-optimal motion planning," *Int. J. Robot. Res.* **33**(1), 18–47 (2014).
51. A. Dobson and K. E. Bekris, "Improving sparse roadmap spanners," *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (2013) IEEE.

Cite this article: M. Shahabi, H. Ghariblu and M. Beschi (2022). "Comparison of different sample-based motion planning methods in redundant robotic manipulators", Robotica **40**, 3104–3119. https://doi.org/10.1017/S026357472200008X

https://doi.org/10.1017/S026357472200008X Published online by Cambridge University Press