IEEE Access
practical innovations : open solutions

Received December 12, 2013, accepted January 22, 2014, date of publication January 24, 2014, date of current version February 4, 2014.

Digital Object Identifier 10.1109/ACCESS.2014.2302442

Sampling-Based Robot Motion Planning: A Review

MOHAMED ELBANHAWI (Student Member, IEEE) AND MILAN SIMIC
School of Aerospace, Mechanical and Manufacturing Engineering, RMIT University, Melbourne 3083, Australia
Corresponding author: M. Elbanhawi (mohamed.elbenhawi@rmit.edu.au)

The work of M. Elbanhawi was supported by the Australian Postgraduate Award and the Research Training Scheme.

ABSTRACT Motion planning is a fundamental research area in robotics. Sampling-based methods offer an efficient solution for what is otherwise a rather challenging dilemma of path planning. Consequently, these methods have been extended further away from basic robot planning into further difficult scenarios and diverse applications. A comprehensive survey of the growing body of work in sampling-based planning is given here. Simulations are executed to evaluate some of the proposed planners and highlight some of the implementation details that are often left unspecified. An emphasis is placed on contemporary research directions in this field. We address planners that tackle current issues in robotics. For instance, real-life kinodynamic planning, optimal planning, replanning in dynamic environments, and planning under uncertainty are discussed. The aim of this paper is to survey the state of the art in motion planning and to assess selected planners, examine implementation details and above all shed a light on the current challenges in motion planning and the promising approaches that will potentially overcome those problems.

INDEX TERMS Planning, sampling, randomization, RRT, PRM, path, motion, autonomous robots.

I. INTRODUCTION

Autonomous robots are characterized by their ability to execute tasks deprived of any human intervention. Decision making requires full or partial knowledge of the surrounding environment, or workspace, in which the agent is operating. Recent advances in sensor technology have enabled the use of reliable multisensory perception systems [1]–[5]. Uncertainty in the perception stage leads to accumulated localization errors [6]. Processing of collected data and accounting for errors is essential for accurate mapping and localization [7]. The planning stage involves devising a collision free strategy from the current location, or configuration, to a desired goal location, or configuration. The current configuration is estimated in the localization stage whereas a behavioral planner can provide a goal configuration [8], [9] (the notion of a configuration will be discussed later). Path planning is a purely geometric process that is only concerned with finding a collision free path regardless of the feasibility of the path. Kinodynamic planning, on the other hand, considers the kinematics and dynamics of the robot. Once a path is specified the final procedure is motion control or execution [10]–[12]. The full potential of autonomous vehicles is yet to be fully exploited in enriching human lives. Nevertheless there exist several applications such as self-driving cars, forklifts, mining trucks, unmanned aerial vehicles (UAV), military drones, cleaning mobile robots, planetary rovers, rescue robots and many more.

A. PATH PLANNING

Planning is not only one of the fundamental problems in robotics [13]–[15], it is perhaps the most studied [16]. Early efforts to develop deterministic planning techniques showed that it is computationally demanding even for simple systems [17]. Exact roadmap methods such as visibility graphs [18]–[20], Voronoi diagrams [21], [22], Delaunay triangulation [23], adaptive roadmaps [24] attempt to capture the connectivity of the robot search space. Cell decomposition methods, in which the workspace is subdivided into small cells, have been applied in robotics [25]. Search algorithms such as Dijkstra [26] and A* [27] find an optimal solution in a connectivity graph, whereas D* [28] and AD* [29] are tailored to dynamic graphs. The use of graph search methods involves discretization of the workspace and their performance degrades in high dimensions. The work in [30]–[32], generates state lattices using motion primitives and combines them with graph search algorithms but, it still suffers from undesirable discretization. Efficient discretization can be achieved on the expense of completeness and high-resolution discretization is computationally expensive. The emergence of novel computational methods inspired their use in path

planning. Methods such as Fuzzy Logic Control [33]–[35], Neural Networks [36], Genetic Algorithms [37], [38], Ant Colony Optimization [39] and Simulated Annealing [20] have all been applied in robot path planning. Khatib [40] proposed a potential field method such that artificial forces repelled the robot away from the obstacles and attracted it towards the goal position. Potential fields were also applied for mobile robots in [41], however they suffered from falling into local minima and performed poorly in narrow regions [42]. Sensor based reactive planning methods have been proposed [43]–[45]. Control based methods require formulating accurate models for the robot and the environment [46], [47], which can be a rather daunting task.

B. RANDOMIZATION AND SAMPLING IN ROBOT PLANNING

Sampling based planning (SBP) is unique in the fact that planning occurs by sampling the configuration space (C-space). In a sense SBP attempt to capture the connectivity of the C-space by sampling it. This randomized approach has its advantages in terms of providing fast solutions for difficult problems. The downside is that the solutions are widely regarded as suboptimal. Sampling based planners are not guaranteed to find a solution if one exists, a property that is referred to as *completeness*. They ensure a weaker notion of completeness that is *probabilistic completeness*. A solution will be provided, if one exists, given sufficient runtime of the algorithm (in some cases infinite runtime).

Sampling based planning is by no means a novel concept in robotics [48]. It was proposed to overcome the complexity of deterministic robot planning algorithms for a robot with six degrees of freedom. The use of random computations to solve otherwise rather difficult problems, have been immensely successful [49], [50]. Both sampling based planners and the success of random computations inspired the development of the Randomized Potential Planner (RPP) [51]. RPP used random walks to escape local minima of the potential field planner. Later on, a planner based entirely on random walks, with adaptive parameters, was proposed [52].

The work of Barraquand and Latombe [51] paved the way for a new generation of motion planning algorithms that employ randomization. Some of these planners are listed in Table 1. Perhaps the most commonly used algorithms are Probabilistic Roadmap Method (PRM) [53]–[55] and Rapidly-exploring Random Trees (RRT) [56]. Several other algorithms were developed at the same time that outperformed RPP. The intuitive implementation of both RRT and PRM, and the quality of the solutions, lead to their widespread adoption in robotics and many other fields.

PRM implements two main procedures to generate a probabilistic roadmap. A learning phase occurs first, where the C-space is sampled for a certain amount of time. The samples, or configurations in the free space, are maintained while those in the obstacle space are discarded. This is followed by a query phase where the start and goal configurations are defined and connected to the roadmap. Roadmaps are sometimes referred to as forests, as an analogy to trees in RRT. As a result of maintaining the roadmap and specifying start and goal configurations in a subsequent stage, PRM is able to solve different instances of the problem in the same environment. It is referred to as a multi-query planner. Planning time is invested in sampling and generating a roadmap so that queries are solved quickly. Initially developed for articulated robots [53]–[55]. PRM has been extended for non-holonomic car-like robots [57]. It was shown that PRM is probabilistically complete [58], [59].

RRT represents another category of sampling based planners, which are single-query planners. A tree is incrementally grown from the start configuration to the goal configuration, or vice versa. A configuration is randomly selected in the configuration space. If it lies in the free space, a connection is attempted to the nearest vertex in the tree. For single query problems, RRT is faster compared to PRM. It does not need to sample the configuration space and construct a roadmap i.e. learning phase. RRT was shown to be probabilistically complete [60].

Expansiveness was proposed as a measure of the number of neighboring nodes to any nodes [61]. It is used as an indication whether a node will be useful in expanding the search tree. Expansive space trees (EST) were developed based on that proposed measure. Unlike RRT where sampling is uniform [56], EST employs a function that sets the probability of node selection based on neighboring nodes.

Ariadne's clew is planner that builds a search tree [62], similar to EST and RRT, to explore the configuration space. The difference in this algorithm is the connection of the randomly selected node. It attempts to connect a node that is furthest from existing nodes. This heuristic is employed to increase the exploration rate of the algorithm. Unlike RRT where the implementation is intuitive by connecting the closest node, a genetic algorithm is used to select the node for expansion [62].

Sampling based planners have been successfully implemented in a variety of fields aside from robotic applications. This is a testament to the generality of the proposed algorithms and their ability to solve difficult and constrained problems. Interestingly, the two fields in which sampling based planning is used are digital animation and computational biology [16]. In digital animation, agents are constructed out of triangular meshes and paths are planned using sampling based planners such as RRT [63] or PRM [64]. A Gaussian-process based Spline-RRT was used to guide a UAV to explore an unknown environment [65]. In computational biology, molecules and proteins are modeled as articulated bodies and sampling based planners are used to simulate protein folding and protein-ligand interactions [66], [67]. EST was used in architectural design to evaluate accessibility of constrained and narrow areas [68]. Medical needles [69] and, deformable objects [70] sampling based motion planning frameworks, have been developed. Several researchers investigated the use of RRT in non-linear control applications such as pendulum control [71]. Apart from simulation based

planning, the first real life applications were reported in multi-robot competitive dynamic environments [72]. Ever since, welding multi-degree of freedom (DOF) robots [73], industrial robots [74], domestic robots [75], [76] and urban self-driving vehicles have used sampling based planning [77], [78].

TABLE 1. Main SBP algorithms.

Planner	Reference	Structure	Remark
RPP	[51]	Combined with Potential Field [40]	Randomly escapes local minima
PRM	[53-55] [57]	Roadmap	Samples C-space, build roadmap and processes multi-query
RRT	[56], [60]	Tree	Randomly samples C-space and incrementally grows tree
Ariadne's Clew	[62]	Tree	Connects node that is furthest away from other nodes
EST	[61]	Tree	Connects node that has more probability of expanding the search

C. CONTRIBUTION

It must be stated that sampling based planning reviews exist in literature. Both surveys [66], [67] focus on RRT and PRM for computational biology and physics-based simulation and modeling. The review papers [16], [79] and the survey by Tsianos, et al. [80] are considered outdated. A significant body of work exists after their publication. Researchers have since evaluated some of the claims and open research questions. Recommendation for planners implementation are proposed [81] and a benchmarking software is presented [82] but they do not survey recent research in the area and present only a handful of planners. Recently, LaValle [83], [84] published outstanding tutorials, which, by no means, can be considered reviews.

In this study we present a survey on state of the art sampling based planners and their applications. The planners are decomposed into different primitives and then differences and similarities between planner's primitives are exposed. We review some of the parameters for selected sampling based planners and, optimal planners, and provide recommendations for implementation. This highlights the importance of parameters and heuristics in sampling based planners and evaluates some of the claims made by researchers. A particular emphasis of this research are recent direction in planning such as optimal planning, real time kinodynamic planning and planning (replanning) in dynamic environments and under uncertainty.

We have introduced the paradigm of robotic planning and highlighted some of the important classical, sensor-based, control based and sampling based planners in section I. The remainder of this paper is arranged as follows, section II an overview of sampling based planners and a formal description of the planning problem are provided. Methods to improve solutions and performances of sampling based planners are presented and some are evaluated using simulations in segment III. We present the problem of kinodynamic planning in segment IV. Optimal planning algorithms are presented and evaluated in segment V. The problem of planning under uncertainty in dynamic environments is then presented in segment VI. The study is finally concluded in segment VII.

II. SAMPLING BASED PLANNING OVERVIEW

SBP is treated as a black box that returns a feasible, collision free path once information about the start and goal configurations is provided, as shown in Fig. 1. In a hierarchical overview of motion planning for autonomous robots, SBP lies between a high-level behavioral planner that specifies global goals and a low-level controller that plans the execution of path.

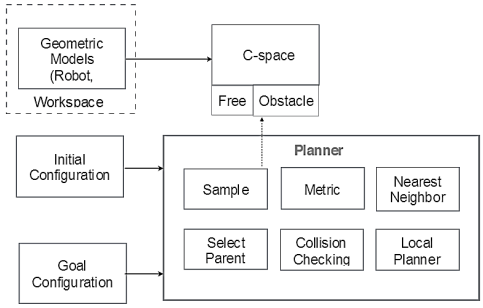

FIGURE 1. A general sampling based planner.

A. PROBLEM DEFINITION

In order to define the motion planning problem some concepts must be introduced. SBP operate, mostly, in the configuration space (C-space). It is the space of all possible transformations that could be applied to a robot. Lozano-Perez [85] introduced the concept of C-space planning to simplify complex planning scenarios in the workspace of the robot. Free space, C_{free}, and obstacle space, C_{obs}, are the two regions within the C-space, C. This prevents the need to explicitly define obstacles. The robot can be only represented by a configuration, q, at any instance. The configuration, q, has equal dimensions as the C-space. Common terminology to describe configurations, such as *nodes, samples, or landmarks,* will be used interchangeably in this study. A sequence of consecutively connected configurations represents a path, P.

Start, q_{start}, and goal, q_{goal}, configurations are the inputs to the motion planner. The problem is to find a collision free path, P_{free}, which connects q_{start} to q_{goal} A path is considered free if its entire configurations lie in C_{free} and their connecting paths do not intersect C_{obs}.

B. PRIMITIVES

It is essential to introduce the constitutions of any SBP algorithm prior to introducing the different planners. Even though these primitives are found in most planners, their

implementation differentiates the planner. Variants of each of these primitives will be thoroughly discussed in section III along with their effect on the performance of the planner.

1) *Sampling:* This procedure is used to select a configuration, randomly, or quasi-randomly, and add it to the tree or roadmap. As mentioned earlier, the samples can be either in the free, or obstacle configuration space. It can be considered as the core of the planner and the main advantage of SBP over other techniques.
2) *Metric:* Given two configurations q_a and q_b, this procedure returns a value, or cost, that signifies the effort required to reach q_b from q_a. It is important that it is truly representative of the effort, or time-to-go between both configurations. Otherwise highly suboptimal solutions will be returned.
3) *Nearest Neighbor (NN):* It is a search algorithm that returns that closest node(s) to the new sample. The value is based on the predefined metric function. Some papers refer to it as proximity search or near vertices.
4) *Select Parent:* This procedure selects an existing node to connect to newly sampled node. That existing node is considered parent. RRT selects the nearest node as the parent. PRM connects the sample to several nodes within its neighborhood. On the other hand, EST selects a parent node to randomly extend based on its neighboring nodes. Ariadne's clew selects a parent node for extension using a genetic algorithm.
5) *Local planning:* Given two configurations q_a and q_b, this procedure attempts to establish a connection between them. It is intuitive to employ straight-line paths. For most robotic systems this is not a feasible plan due to kinematic or dynamic constraints.
6) *Collision checking (CC):* It is mostly a Boolean function that returns success, or failure, when connecting two configurations. A connection is successful, if it does not intersect C_{obs}.

C. ALGORITHMS

Algorithms for PRM and RRT are presented here as introduced in [55] and [56]. They are the main algorithms used in SBP. It must be noted that configurations may be referred to, using common SBP literature terminology, as nodes or milestones, throughout this study.

1) RRT

- The search is initialized from q_{start}.
- A node, q_{rand}, is selected from the C-space using the sample procedure, as shown in Fig. 2(a).
- q_{rand} is discarded, if it is in C_{obs}.
- Using Nearest Neighbor search q_{near} is returned according to the metric, as shown in Fig. 2(b).
- The local planner is used to connect q_{rand} and q_{near}. The planner may return q_{new}· q_{rand} may not be reachable, as shown in Fig. 2(c). If q_{rand} is not reached, it is discarded.
- Collision checking is performed to ensure the path between q_{near} and q_{new} is collision free. If path is collision free q_{new} is added to the tree as shown in, as show in Fig. 2(d).
- The search terminates when $q_{\text{new}} = q_{\text{goal}}$, a number of iterations is exceeded or a specified time period are exceeded.

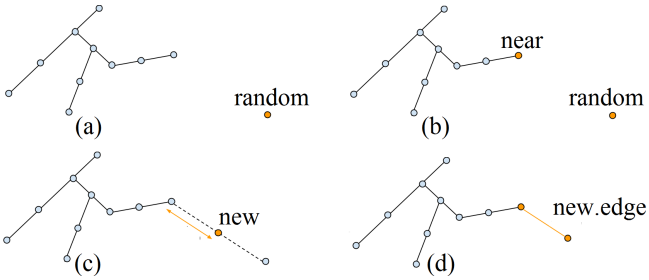

FIGURE 2. The procedure of extending RRT.

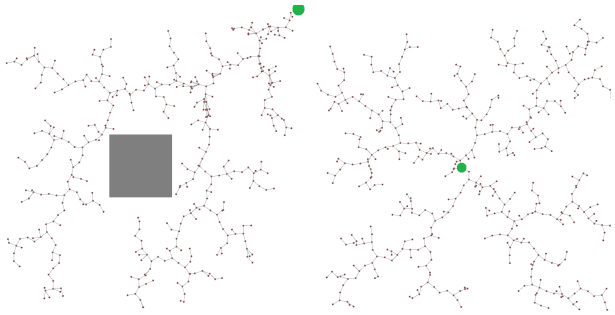

FIGURE 3. RRT exploring free space (right) and environment with one obstacle (left) after 500 iterations. The root of the trees in both cases is shown as green bold green circle, in top right corner (left) and center (right).

The ability of RRT to explore free space in presence and absence of obstacles is illustrated in Fig. 3. This property is often referred to as the Voronoi bias of RRT. As a result of uniform sampling, the planner is more likely to select samples in larger Voronoi regions and the tree is incrementally and rapidly grown towards that free space.

2) PRM

Firstly, a roadmap is built in the learning phase,

- A node, q_{rand}, is selected from the C-space using sample procedure.
- q_{rand} is discarded, if it is in C_{obs}.
- Otherwise, q_{rand} is added to the roadmap.
- Find all nodes within a specific range to q_{rand}
- Attempt to connect all neighboring nodes using local planner to q_{rand}.
- Check for collision and disconnect colliding paths
- This process is repeated until a certain number of nodes have been sampled.

A typical roadmap, built in the learning phase, is shown in Fig. 4. In the query phase the start and goal configurations are connected to the roadmap. A graph search algorithm is then

used to find the shortest path through the roadmap between start and goal configurations.

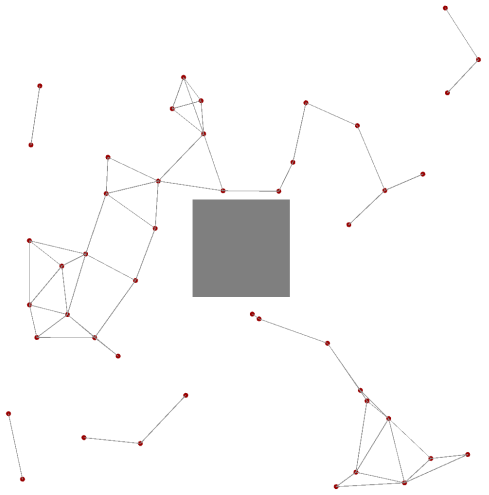

FIGURE 4. Roadmap built in the PRM learning phase.

III. PARAMETERS AND HEURISTICS

Sampling based planners consist of a number of primitives with varying parameters. A significant portion of research in SBP is dedicated to designing algorithms with smart heuristics and parameters.

The aim of these improvements is generally twofold, reducing algorithm run time and cost of solutions. In this section SBP variants are categorized and surveyed. SBP are rather sensitive to their implementation and some emphasis must be placed to selecting the correct parameters [86]. Sucan and Kavraki [81] highlight the importance of parameters and argue that the implementation details are often not mentioned when SBP are presented. Motivated by the reliance of RRT on heuristics, Randomized Statistical Path Planning (RSPP) applies machine learning to actively adjust the planners parameters while the algorithm is running [87]. In this section, a number of implementations and parameters are tested using simulations in various scenarios.

A. SAMPLING STRATEGIES

Sampling is the core of the SBP. It is the process through which the planner is able to extend and explore the C-space. Initially, PRM and RRT were proposed with uniform sampling schemes [55]–[57]. This can be considered as a drawback because the planner has a high probability of sampling a node from a wide region unlike a narrow free region. This is a result of all configurations have uniform probability of being sampled and narrow regions have less free configurations. Another drawback of uniform sampling is not capturing the true connectivity of the environment. The following sampling strategies have been suggested as means to overcome those shortcomings,

- Medial axis: Sampling probability is increased around the medial axis (Voronoi graph) to guide the generation of a roadmap that fully captures the shape of the C-space [88]–[90].
- Boundary: Forcing sampling towards the boundary of obstacles, as opposed to free space, was proposed in [54].
- Gaussian: Similar to boundary sampling, this strategy increases the probability of sampling around obstacles. Nodes are expanded using an adaptive probability based on obstacle and collision data [91].
- Bridge-test: This overcomes the weakness of SBP in narrow regions. The strategy uses a short segment with two configurations and their midpoint [92]. If the two ends are in C_{obs} and their midpoint is in C_{free} then a narrow region has been identified.
- Hybrid: Combining two sampling strategies, narrow passage (bridge-test), and uniform sampling. This lead to an increase density in narrow regions and still maintaining randomization which is advantageous in solving difficult problems [93]. Medial axis and narrow sampling are combined to better capture the environment connectivity [94].
- Visibility PRM [95]: A non-uniform sampling method. Sampling is performed in visibility regions. It decreases the number of nodes maintained in the roadmap while maintaining the same coverage.
- Goal Biasing: It may not be considered as a sampling strategy however biasing is mentioned here as it is used to replace sampling strategy for an interval at some planning stage. Biasing attempt to greedily connect the goal configuration to the current tree [96]. Biasing is recommended, between 1-10 every 100^{th} iteration, to maintain randomization in sampling [13], [84].

The effect of sampling on the performance of SBP is still an open research question. The experimental results presented by Lindemann and LaValle [79], Geraerts and Overmars [86], [97] show that sampling has no effect on the performance of planners. It also shows that there is no single sampling strategy that outperforms the others in every scenario.

B. GUIDING THE EXPLORATION

The motivation behind the attempts to guide the search is that RRT expansion is more prone to fail if the node is near and obstacle (boundary node). A simple approach is to attempt to limit the sampling domain to the visibility region, which is difficult to compute. Dynamic-Domain RRT (DD-RRT) limits the sampling domain of boundary nodes to a small ball of a predetermined radius as an alternative to the visibility region [98]. Adaptive Dynamic Domain RRT (ADD-RRT) limits the domain to a ball, whose radius changes according to the extension success rate of each boundary node [99].

Unlike ADD-RRT and DD-RRT, Utility-RRT influences the direction and length of extension, not the sampling domain. A utility function evaluates the direction of expansion and the selected node [100]. Utility functions are computed based on the success rate of the node and previous

direction of expansion. Obstacle Based RRT (OB-RRT) gathers data from obstacles and selects predetermined growth directions [101]. Utility-RRT outperforms both ADD-RRT and RRT [100], OB-RRT has only been benchmarked against RRT. OB-RRT relies on obstacles models consisting of triangles. No discussion is provided whether this method would extend to other representations.

A novel categorization divides motion planners into exploring and exploiting planners [102]. SBP presented here perform guided exploration. On the other hand, artificial potential field algorithms and wave front decomposition [103] exhibit purely exploitive behavior. Exploring/exploiting tree (EET) balances both behaviors based on successful expansion of the tree [102]. It attempts to use purely exploitive behavior to provide fast solutions for subproblems and leverages exploring behavior of SBP when the planner fails.

Several adaptive sampling strategies have been proposed. Significant reduction in planning time for a non-holonomic UAV is achieved by increasing the density of sampling around the goal region once the tree approaches it [104]. A high level planner modifies the sampling domain to influence the behavior of a self-driving car by manipulating the Closed Loop RRT (CL-RRT) growth [77]. An estimation model predicts the probability of a sample, to optimize the solution and adapts the sampling strategy accordingly, to direct the search towards lower cost regions [105]. Collision information is used to adapt sampling when building a roadmap in real time [106].

C. METRICS

Metrics are used to indicate the *cost* or *time to go* between two configurations. PRM and RRT are reliant on metrics for extending their search. Choosing an accurate metric is arguably as difficult as the motion problem itself [13]. It is of the utmost importance that metrics provide a good estimation, not necessarily exact, of the cost between two configurations. Metrics can be called multiple times during the planning procedure so it must be easily computed. A theoretical analysis of path quality measures in a plane is presented in [107].

EST and Guided Expansive Space Trees (GEST) [108] select nodes for expansion based on their neighboring nodes. Path Directed Subdivision Trees (PDST) [109] and KPIECE[1] [110] select nodes for expansion based on their coverage, to ensure that expansion is not wasted on already explored areas. These planners reduce their dependency on metrics.

Amato, et al. [111] experimentally studied the effect of different metrics on PRM and reported that the best performance was obtained by using a weighed Euclidian metric. This metric accounted for rotation as well as linear Euclidian distance. Similarly, accounting for rotation using Euler angles, or Quaternions, proved to be advantageous when planning with RRT in 3D [112].

[1] Kinodynamic Planning by Interior Exterior Cell Exploration (KPIECE)

Non-holonomic vehicles such as car-like ground robots or UAV with upper-bounded curvature are common robotic platforms. Euclidian metric is a poor choice for those vehicles since two configurations that are physically close may require complex maneuvering to reach (see subsection III-G for discussion on local planning). Calculating the true cost involves expensive computations which is infeasible given the frequency of the metric function usage during planning. SRRT uses a Euclidian distance to calculate the closest k-neighbors, where k is a positive integer, and then connect to the one with the smaller real distance [104]. Another approach overestimates the distance when the Euclidian distance is less than the minimum turning radius, indicating that a complex maneuver might be needed [113]. Manipulability was proposed as a metric for articulated robots to signify the ease by which the robot can reach a certain configurations, especially that articulated have redundant configurations [114].

As a substitute for purely relying on a distance metric to select the suitable node for expansion, the failure rate of previous node expansions is factored in the selection metric, an approach, that is often referred to as Resolution Complete RRT (RC-RRT) [115], [116] and was adopted in [117]. This prevents wasting planning time on nodes that are bound to fail simply because of their low metric value. RRT-Blossom choses an expansion node similarly [118]. However it proceeds to expand the node in all directions and removes nodes that are close to nodes already in the tree. This approach has a drawback of discretizing the control space, which is one of the strengths of RRT, as it operates in a continuous space. Discretizing the control space has been shown to improve planning for some nonlinear systems [119], [120]. It is yet to be evaluated for differentially constrained robotic planning.

The costs that arise between two configurations simply account for the effort needed to drive the robot from one to the other. All the previously mentioned approaches assume a uniform cost C-spaces, aside from heuristic method presented in [121]. Non-uniform costs are used to signify non-uniform rough terrain [122], estimated uncertainty [123], or can be user defined to bias the plan towards preferred regions [124]. Transition-RRT (T-RRT) [125] were proposed to handle non-uniform cost C-space, referred to as cost maps. It provides an adaptive criterion, referred to as transition test, which prevents transitioning into costly regions based on the cost differences between parent and child nodes.

D. COLLISION CHECKING

One of the main properties of SBP is that obstacles in the environment are not explicitly defined. Planning generally takes place in the C-space, which is separated into C_{free} and C_{obs}. This approach requires a module, which provides information on whether a path collides with any obstacle. Since the goal of SBP is to create collision free paths in the C-space, it stands to reason that collision checking (CC) will be called several times during planning. Some experiments show that more than 90% of planning time is spent processing CC queries [93]. It can be

noticed by from any SBL that most connections are collision free.

Several planners use CC as a feedback mechanism to guide the search [72], [98], [99], [108], [115] adapt the sampling strategy [105], [106], or improve the connectivity of the environment [92], [93], [126]. Proximity Query Package (PQP) is commonly using for CC [127]. An experimental comparative analysis shows that other packages outperform PQP [128].

Lazy planning algorithms have been proposed to delay collision checking until it's needed [129]–[131]. These algorithms will check the collision only once a path is found. Once collision is detected the colliding segment is removed and planning is continued. Another approach is to decrease the reliance of expensive CC. The distances between free configuration and C_{obs} are maintained and similarly obstacle configurations and C_{free}. These distances are used to infer whether a new configuration or, new path segment is colliding and decrease the reliance on CC [132].

Single-query Bidirectional Lazy (SBL) is a planner that not only delays planning but it also performs CC in regions that are more likely to collide [133]. The CC algorithm in SBL is based on four observations:

1) A small fraction of all samples is in the final path (around 0.1%),
2) Incrementally checking the path is computationally expensive, especially when no collision is detected, as the entire path must be checked,
3) Short connections are more likely to be collision free between two configurations in C_{free}, as shown in Fig. 5(b),
4) Collision is more likely to be in the midpoint between two configurations, as shown in Fig. 5.

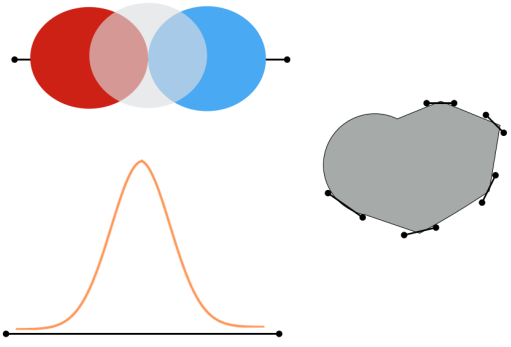

FIGURE 5. Illustration of the observations made by Sánchez and Latombe [133]. (a) The midpoint of a colliding path between two free configurations is more likely to be in C_{obs} (b) It is difficult to have a colliding path between two free configurations that are separated by a short distance. The collision is still more likely to be towards the midpoint of the short line.

A collision checking algorithm is employed by SBL based on the observations made in [133]. Naive CC is performing incremental checking at some interval from one end to the other along a path, shown in Fig. 6(a). SBL checks the midpoint between two configurations dividing the path into two parts, shown in Fig. 6(b). If the midpoint is free, the midpoints of the two parts are checked. This process is continued until a certain resolution is reached.

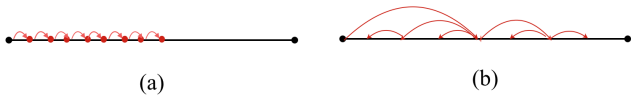

FIGURE 6. Red arrows connote a CC query between two configurations that are connected by the black solid line (a) Naive incremental collision checking (b) SBL midpoint collision checking.

E. HEURISTICS

In this section we introduce some methods that have been shown to refine the solution cost or planning time of SBP. It must be noted that there are no theoretical guarantees to those claims. However, these planners have been shown to work well in various situations. We will provide some discussions about the strengths and shortcoming of those tactics.

RRT-Connect [96] and SBL [133] use two trees to perform bidirectional search. One tree is rooted at the start, whereas the other is at the goal. The search is complete when the two trees are connected. This approach provides significant improvements in the search efficiency, which is illustrated in Fig. 7. Triple RRT [134] generates two trees from start and goal configurations and one tree from a narrow region which is identified using the bridge test. Similarly, Multiple RRT are generated from all narrow regions, in the free space that are identified using the bridge test [135]. A problem arises when attempting to connect two trees for differentially constrained systems where the local planning is not a simple straight line, resulting in what is known as a boundary valued problem [136]. Methods to overcome this problem will be discussed in section IV.

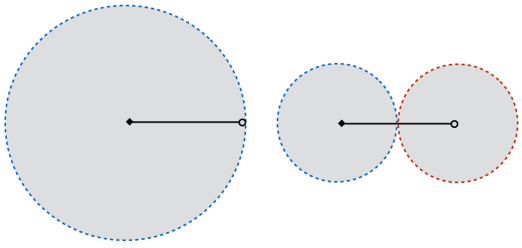

FIGURE 7. Unidirectional search coverage area (left) and bidirectional search (right). The search is started from the diamond shaped configuration and the final configuration is circle-shaped. Employing two search trees is more effective since less area is searched to find the solution.

It can be seen that using NN search to connect the sampled node to the nearest node does not necessarily improve the path cost. A k-near RRT employs NN search to find the nearest k nodes, where k is a positive integer [121]. The path is evaluated for all the k-nearest nodes and the node with the best solution is connected to improve the overall solution. The drawback of this approach is computational overhead as, NN search is called, and metrics are evaluated more frequently.

An alternative to relying on NN search is evaluating the path towards a candidate node with all nodes in the tree [137].

Anytime RRT deals with lack of computational time for path improvement by generating an initial suboptimal solution [138]. The tree is then stored and the rest of the time is used to attempt to improve every solution by a predetermined bound (generally 5-10%). This is achieved by applying a node selection strategy. If the underestimated, lower-bound, path cost through the candidate node is less than the current path cost it is deemed ''promising'' and added to the tree. Waypoint caches were originally proposed for real-time planning are also used to guide replanning with anytime RRT [139]. It is explicitly remarked that Anytime RRTs improve the path within the given planning time, however they provide no guarantees on reaching an optimal solution under certain criteria and time constraints. This property is known as asymptotic optimality and will be discussed in section V.

F. POST PROCESSING

A major drawback of SBP is their widely regarded suboptimal paths. This is as a result of the arbitrary approach used in sampling and heuristics that are employed to speed up the search. Whereas some methods attempt to guide to improve the path quality during the search process [121], [138], the algorithms in this section proceed to smooth and modify the path after planning is complete. Post processing is illustrated in Fig. 8. The original path is shown as a thin line, the dotted line is the trimmed path, and finally the bold line shows the smooth curved path.

Simply inspecting subsequent nodes and removing redundant nodes achieve path shortcutting, or tree pruning. An efficient algorithm removes redundant nodes in one dimension at a time and provides some clearance by moving the path towards the medial axis [140].

Smoothing techniques rely on using a curve to interpolate or fit the given waypoints. These methods are not limited to SBP but have been used in various scenarios and with planners. Methods such as cubic polynomials [141], quintic polynomials [142], [143], Bezier curves [144]–[147], B-splines [20] and Clothoids [148] have been all applied for path smoothing. An study shows that Bezier and B-splines are well suited for robotic planning and B-splines were shown to be more effective in replanning situations in dynamic environments [149].

Hybridization graphs (H-graphs) are constructed by coalescing multiple RRTs and attempting to optimize the solution [150]. This work is based on the observation that RRTs are globally suboptimal, conversely some local optimality exists. It is hoped that the locally optimal components of different trees can be combined to achieve global optimality. Hybridization is used with trees generated using the same planner. No studies have been performed on the effect of using trees generated with different parameters. The effect of having a portion of trees rooted at the start, others at the goal and utilizing bidirectional trees are prospects, which are yet to be investigated within the hybridization framework.

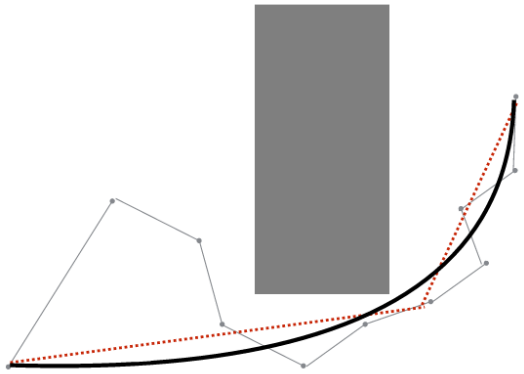

FIGURE 8. An illustration of post processing. The original path is highly suboptimal (grey thin line). Redundant nodes are removed and the rest are connected to provide a shortcut path (red dotted line). Smoothing techniques are then employed to fit a curve through the short path (black thick line).

Post processing, as is the case with any SBP stage, is limited by an amount of time. Alternating between hybridization and smoothing within the given time-frame have been shown to be effective and computationally efficient [151]. Path Deformation Roadmap (PDR) extends on the notion of Visibility PRM by removing redundant paths that can be deformed into other existing paths [152]. Maintaining a compact deformable roadmap facilitates post processing as various paths between two roadmaps can be easily obtained.

Regardless of the effectiveness of these approaches, post processing does not regulate the impractical attempts to expand nodes towards suboptimal regions. It only proceeds to optimize the path at a later stage. Planning time is wasted in both the search and the optimization stages. A more efficient strategy would be to explicitly consider path quality during planning.

G. LOCAL PLANNING

Steering functions are employed to connect configurations, or landmarks, in SBP. Intuitively, a straight line joining both configurations may be proposed. In the case of differentially constrained robots, or non-holonomic robots this may not be feasible. A viable approach is to model the robot system and sample the control space for a certain period of time. However, it must be noted that a tradeoff exists between computational efficiency and accuracy when using numerical integration. Kinematic model for a car-like vehicle is often represented using the bicycle model as follows in Eqn. (1), where x and y are the vehicle coordinates, θ is it's orientation, $\emptyset$ is the steering angle, v is the linear velocity and L is the distance between the front and back wheels, as illustrated in Fig. 9. Non-holonomic planning is a thriving area of research [153], which can be combined with SBP to provide effective planning techniques.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\emptyset} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \tan\theta/L \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \dot{\emptyset} \qquad (1)$$

Dubin's path [154] and Reeds and Shepp's [155] are commonly used for non-holonomic vehicles that are bound by a minimum turning radius [57]. They combine circular arcs and straight lines to generate optimal paths, however the curvature of the path may not be continuous. Curvature continuous paths were proposed using Clothoids [148], [156]. Clothoids have no closed form representation and thus provide computational challenges to synthesize in real time [157]–[161]. Bezier curves were proposed for smoothing [144], [145] and then they were used for local planning in SRRT [65], [104]. B-splines were proposed for planning and replanning in dynamic and unknown environments [149].

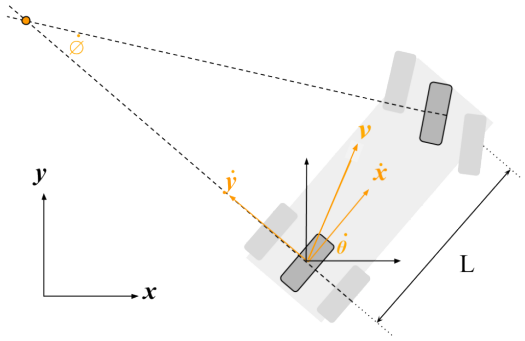

FIGURE 9. Illustration of bicycle model of car-like vehicle rotating, the instantaneous center of rotation (ICR) is shown as circular node.

In order to improve the connectivity of PRM roadmap Delaunay triangulation have been used for local planning [162]. Toggle PRM initially implements a straight line connection. If connections fails, it attempts to establish a connection from the same node in different directions [126]. PRM is combined with RRT or EST as local planners to take advantage of both planners' strengths in solving complex queries [163]. PRM samples milestones and maintains roadmap while single-query motion planner attempts to connect milestones. The planner, formalized as Sampling-Based Roadmap of Trees (SRT), was shown to be more efficient than using a standalone PRM, RRT or EST [164].

H. SIMPLIFYING THE PLANNING PROBLEM

It is often useful to limit the search space dimensions as a means to facilitate the planning process. Motion primitives are often used for highly redundant robots. These robots can solve a single query, i.e. reach a pose, in a various configurations [165], [166]. Certain planning dimensions are disregarded by constraining the motion of the robot to a specific manifold or moving the planning problem into a lower dimensional space that is more relevant the task [76], [167]–[169].

Maneuver based planning was proposed, in which stable trim-trajectories are known *a priori* and used to connect nodes [137]. The concept of maneuver based planning have been extended into Maneuver Automata, as alternative to optimal control methods [170]. They consist of a finite set of interconnected motion primitives; the connections are governed by some rules to ensure dynamic feasibility. Atlas RRT [171] projects the highly constrained C-space manifold into overlapping charts which are contained within an atlas to overcome the complexity of C-space introduced by kinematic constraints.

I. IMPLEMENTATION ENHANCEMENTS

It can be argued that most of the research on SBP is focused on theoretical aspects and implementation details are often left out of the discussion [81]. SBP parameters have been shown to have a significant effect on their results [86]. Statistical learning has been used to adaptively adjust parameters [87].

An open-source library has been developed as a common benchmarking tool that limits the effect of implementation parameters [82]. Kd-trees have been used to improve the efficiency of NN-search [172]. Taking advantage of powerful CPUs by parallel processing and running multiple searches have been shown to be effective [164], [173], [174].

J. EXPERIMENTS

In order to illustrate the significance of SBP parameters on the results obtained several experiments are presented here.

1) EXPERIMENTAL SETUP

RRTs are used to solve single queries two-dimensional environments. We highlight the effect of several implementation parameters such as step size used for extending the RRT, the percentage of biasing, k values in k-RRT, bidirectional search. We also test some of the observations used for lazy CC in SBL.

RRT rely on random sampling. As a result, running the same algorithm with the same parameters will produce different solutions. Some solutions can be near optimal, lucky, while others may be grossly suboptimal, pathological cases. Both cases are shown in Fig. 10. There are three obstacles in the environment shown as grey boxes. The goal region is highlighted as the green box, the path is shown in red and the RRT is shown as black lines.

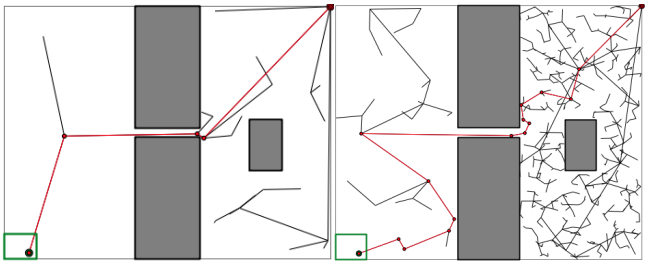

FIGURE 10. Lucky (left) and pathological (right) solutions obtained by running the same algorithm without changing any parameter. This environment is referred to as "narrow" in this study.

Several measures have been put in place to ensure that the presented results are truly reflective of parameter effects. Firstly, any experiment is looped for 54 runs, the best and worst two results are then omitted, and the remaining 50 are

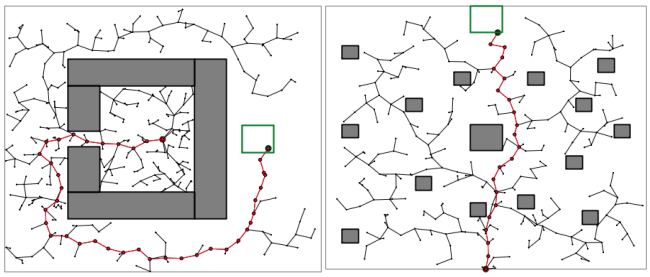

FIGURE 11. Trap environment is shown on the left and cluttered is shown on the right.

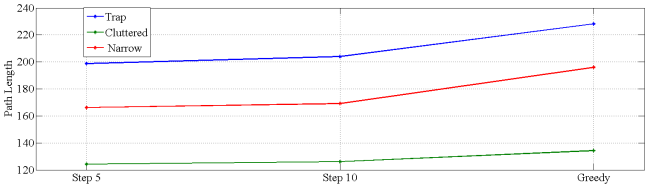

FIGURE 12. Effect of changing the step size of RRT extend on the path cost, a minor implementation detail that has a large effect on the performance.

averaged. All experiments are run on three environments, each with its own challenges. Environment dimensions are 100x100 in all cases, obstacles are grey objects, goal region is a green box, RRT is shown in black, and the final path is highlighted in red. The environments are referred to as narrow, trap and clutter in this study, shown in Fig. 10, Fig. 11 left and Fig. 11 right respectively. All the experiments are implemented in Python.

2) RESULTS AND DISCUSSION

The goal of the experiments in this segment is to highlight the effect of implementation parameters on RRT. It must be noted that the number of explored nodes is used as an indication of the algorithm run time and the cost is the Euclidian distance.

The result of changing the step size of the RRT extension on the path cost is presented in Fig. 12. The step size is tested with 5, 10 and then it is unrestricted. Restricting the step in which the RRT is incrementally grown, maybe counter-intuitive but it generates far better solutions. The planning time saved by unrestricting the step size will be lost in post-processing to improve the solution.

An RRT planner is tested with no biasing, 5% and 10%. This percentage indicates the percentage of planning in which the planner attempts to greedily connect to the goal configuration. The results of biasing are given in Fig. 13. It is expected that biasing will pull the tree towards the goal, leading to decreasing the number of nodes explored. In both, the narrow and cluttered environments, this is true. It is not the case in the trap environment where the tree must first move away from the goal then return to it. Increasing the biasing leads to increased computation.

Biasing is then compared with bidirectional search by generating two RRTs. Results are given in Fig. 14. Bi-RRT provides more consistent improvements across all environments. As previously mentioned, the main drawback of bidirectional RRT is the subsequent BVP, for differentially constrained systems, when attempting to connect two trees.

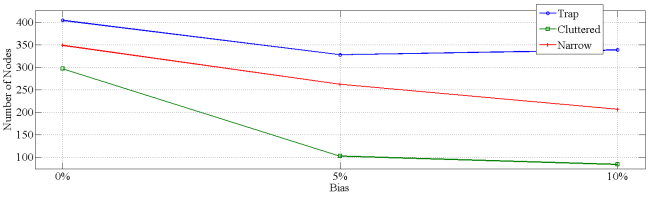

FIGURE 13. Effect of goal biasing on the path cost of the number of nodes explored before a solution is found, which is an indication of the algorithm run time.

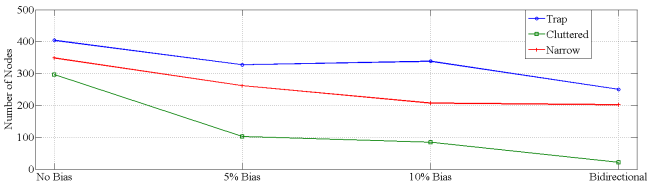

FIGURE 14. Comparison between RRT, Biased RRT and Bidirectional RRT explored nodes before finding a solution.

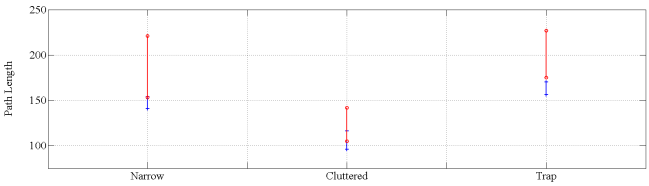

FIGURE 15. The range of the results (path cost) of k-RRT (blue lines) compared to results returned by RRT (red lines). The small variation in the solution returned by k-RRT indicates more consistent performance and reliability.

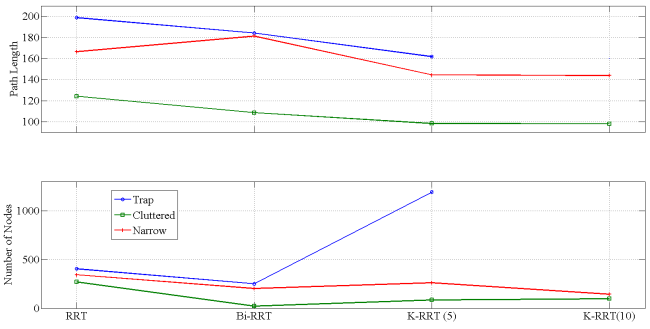

FIGURE 16. Comparing between RRT, k-RRT and Bidirectional RRT in terms of path cost (top) and number of explored nodes (bottom).

Performances of bidirectional RRT and k-RRT, for $k = 5$ and 10, are compared. Path cost and the number of nodes are show in Fig. 16 and the range of the results of using k-near ($k = 5$) is shown in Fig. 15. It can be seen that both cases of k-RRT produce, better solutions than Bidirectional

RRT. However, the computational time needed by k-RRT far exceeds that of Bidirectional RRT and in the case of $k = 10$ the planner fails to find a solution in the trap environment in the specified planning time. Another advantage of k-RRT is the consistency in its results despite its reliance on sampling. This is illustrated by the range of path cost solutions provided by k-RRT ($k = 5$) in comparison to RRT, shown in Fig. 15.

TABLE 2. Path failure rate using lazy collision checking.

Environment	Narrow		Cluttered		Trap	
Step Size	2.5	5	2.5	5	2.5	5
Failure Rate	32%	52%	32%	50%	36%	44%

The motivation behind approaches that employ lazy CC is that a small fraction of most connection collides with obstacles. This observation is consistent with the experiments conducted here. Once a solution is found, CC is performed. If a resulting path collides with obstacles, the planner will either discard the colliding segment or the entire path.

The percentage of infeasible solutions due to collision is often not considered when employing lazy collision checking. If a large number of paths are colliding, it may be more effective to employ an efficient CC algorithm for all connections. The percentage of colliding paths in different environments and under different step size is shown Table 2. As expected as the step size decrease so does the failure rate. However, the path failure rate remains high. It is a question of implementation, whether it is more efficient to employ lazy CC and re-plan the path almost 30%-50% of the time, or constantly employ efficient CC.

K. SUMMARY

Table 3 is a summary the body of work reviewed in this section.

IV. KINODYNAMIC PLANNING

Kinodynamic planning deals with the kinematic, non-holonomic and, or, dynamic constraints imposed on the robotic system or vehicle. The previously presented planners were purely geometric, considering only the feasibility of the path. The term ''Kinodynamic'' has been coined as a synergy between kinematics and dynamics [175]. Deterministic planners were proposed, however, they suffered from high computational costs [175], [176].

In some cases path planning and kinodynamic constraints are decoupled. Traditionally, Planners generate a path that relaxes all kinodynamic constraints. Trajectory modification can be employed to gradually modify the trajectory to obey the constraints. Trajectory modification uses small forces to incrementally alter the course. It has been proposed for non-holonomic [177], [178] and kinodynamic constraints [179]. Trajectory modification has been successfully applied for humanoids [169], car-tests [180] and multi-DOF non-holonomic planning [181]. Discarding kinodynamic constraints during planning may lead to highly suboptimal solutions that involve difficult maneuvers. Worse, the robot may not be able to execute the plan, resulting in

TABLE 3. Summary of SBP parameters and heuristics.

Sampling		
Method	References	Strategy
Medial Axis	[88-90]	Sampling around Voronoi graph
Obstacle	[54, 91]	Sample around obstacles
Narrow	[92]	Sampling in narrow regions using bridge test
Hybrid	[93, 94]	Combining multiple sampling strategies
V-PRM	[95]	Limit the number of nodes only to connect non-visible regions
Biasing	[96]	Attempts to connect to goal region every i^{th} iteration
Guiding the Exploration		
Planner	Reference	Strategy
DD-RRT	[98]	Limit the sampling domain for boundary nodes
ADD-RRT	[99]	Adaptively changes sampling domain for boundary nodes based on history
Utility RRT	[100]	Selects direction and length of exploration based on previous attempts
OB-RRT	[101]	Selects direction of exploration from predetermined sets based on obstacle geometry
EET	[102]	Balance SBP exploring with fast exploitation
CL-RRT	[77]	Sampling domain is changes based on goal specified by high level planner
CE-RRT	[105]	Prediction model is used for adapting the sampling towards low cost regions
CALM	[106]	Adapts sampling based on collision data
Metrics		
References	Metric	
[111, 112]	Weighed Euclidian in 2D and Quaternions in 3D to account for rotation	
[61, 108]	Select node based on probability of expansion not metrics	
[109, 110]	Select node based on coverage of C-space not metrics	
[104, 113]	True, non-holonomic, cost is calculated only when needed, otherwise it is estimated	
[114]	Dexterity of articulated robot is used as a metric	
[115, 118]	Include success rate of expansion for each node to prevent wasting planning time on non-promising nodes	
[125]	Transition test for planning in non-uniform cost maps	
Collision Checking		
Strategy	Reference	Remark
Lazy	[129-131, 133]	Perform CC for path when it is found
Efficient	[133, 174]	Check path midpoints and subsequent midpoints of path segments. Infer collision without calling collision checking module
Feedback	[72, 92, 93, 98, 99, 105, 106, 108, 115, 131, 138]	Use collision information to guide search, adapt sampling strategy and improve connectivity
Heuristics		
Planner	Reference	Remark
Multiple Trees	[96, 134, 135]	Employ multiple trees to improve search efficiency
k-RRT	[121]	Select the node that is more likely to improve path cost
Anytime RRT	[138]	Attempts to improve the path within the given planning time by promising node selection
Post Processing		
Method	Reference	Remark
Shortcutting	[140]	Removing redundant motion and nodes
Smoothing	[141-147]	Curves and splines are employed for smoothing
Hybridization	[150]	Combines multiple solutions
Alternating	[151]	Syndicates hybridization and smoothing
Local Planning		
Method	Reference	Remark
Circle, arcs and lines	[154, 155]	Path is not continuous
Clothoids	[148, 156]	Difficult to compute online
Splines	[104, 149]	Curvature control is challenging
Polynomials	[143]	Difficult to compute online

unrecoverable situations that lead to collision. Considering system dynamics is favorable. For some systems attempting to accurately model all the effects will overcomplicate the model and increase the planning space dimensions and the search complexity.

SBP conducts a search in the C-space by sampling configurations, q, and attempting to extend the search towards those configurations. Kinodynamic SBP operates in the state space, χ, which contains a set of all possible states, x. State space can be considered as C-space augmented with velocities. Subsequently, state space has more dimensions. A state is generally defined as Eqn. (2).

$$x = f\,(q, \dot{q}) \tag{2}$$

There are several issues confronting kinodynamic planning. It is inherently a high dimensional problem, since considering the first derivatives, for the robot configurations, effectively doubles the dimensions of the search space. The state equation of the robotic system must be known, as shown in Eqn. (3). The control space, U, is then discretized. The control, u, that drives the robot as close as possible to the desired state, x, is selected [60]. Similarly, Frazzoli, et al. [137] attempt connections from all nodes in the tree using an optimal controller until the desired state is reached Hsu, et al. [182] select a random state, x, and apply a random control, u, for a fixed time. Eqn. (3) must be integrated, for the time period that control u is applied, in order to determine the local trajectory that joins two states. KPIECE employs a physics simulator to generate the motion trajectory between states [110]. There is a trade-off between the accuracy of the trajectory generation process and its computational efficiency.

$$\dot{x} = f\,(x, u) \tag{3}$$

Planning in a C-space that has narrow corridors, similar to Fig. 10, is one of the challenges in SBP. Kinodynamic constraints limit the motion of the robot, essentially creating narrow passages in the state space. The state space is, traditionally, defined into free and obstacle state space. Another subdivision exist in the free state space, referred to as Inevitable Collision States (ICS), in which the robot will collide with obstacles regardless of the control input applied [183]. This contributes to the complexity of the planning dilemma, as the free state space is narrower. High dimensional planning combined with narrow passages in the free state space leads to slowing down SBP planners. Synergistic combination of layers of planning (SyCloP) is a framework that handles these issues by combining two layers of planners, a discrete and a continuous tree planner [184]. The deterministic layer defines where the SBP planner should start planning and changes the search area if the SBP is deemed stuck.

Defining a metric that evaluates the true cost between two states is another challenging problem in kinodynamic planning. Poor metric selection leads to ineffective planning. Euclidian for instance will identify a state as suitable candidate for extension. However, extension from this state maybe redundant, as it does not expand the search, or may constantly lead to collision. Often a trajectory is generated in such a way to optimize a cost function [60]. Similar selection strategies to the ones proposed in path planning to decrease reliance on metrics have been used in motion planning, such as expansiveness [182], state space coverage [110], [163], [184] and accounting for previous success of expansion [115]. The sensitivity of RRT to metrics is more problematic in differentially constrained kinodynamic planning, as extending procedures are computationally extensive. For some systems it is possible to formulate a pseudo-metric estimate for the true cost by linearization of the system dynamics and quadratization of the cost [185].

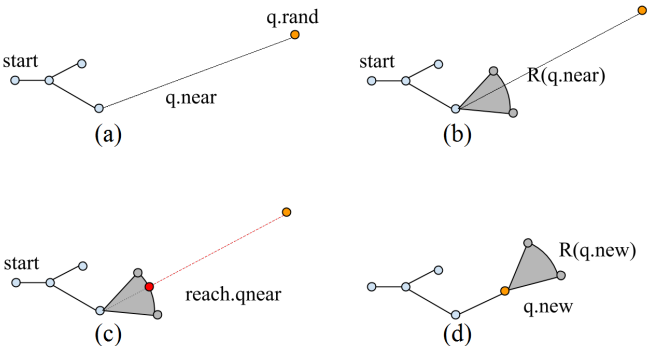

FIGURE 17. Illustrating the RG-RRT extension procedure (a) Select a random node and find the nearest node in the tree (b) Compute the reachability of the nearest node, shown as a grey shaded arc (c) Find the nearest reachable node to the random node, shown as a red node. Compare the distance between the nearest node and the nearest reachable node (red) (d) Extension will only be executed if the reachable node is closer, it is then added to the tree.

Extending the tree requires integrating the equations of motions to obtain the desired trajectory. The reliance on metrics means that several extensions are wasted, as they will not contribute to find a solution. Reachability Guided RRT (RG-RRT) evaluates a reachable set for any node in the tree [186], as shown in Fig. 17(a) and (b). RG-RRT is based on the observation that expansion of the tree is more expensive that sampling for differentially constrained systems. A node is added to the tree, if it's closer to the nearest reachable node than to the nearest node in the tree, as shown in as shown in Fig. 17(c) and (d). Environmentally Guided RRT (EG-RRT) [123] combines two efficient strategies of RG-RRT [186], of adding reachable nodes, and RC-RRT [115], of considering failure and success rate of a node prior to selecting it.

Kinodynamic planning has been limited to simulation based planning applications. Planning time can reach several minutes in some simulation scenarios [60]. Real time kinodynamic planning in state space require exponential planning time [137], [187]. Initial attempts to apply kinodynamic planning in real life situations produced inaccurate results and resorted to a decoupled planning hierarchy where dynam-

ics are handled by another module in a step that followed path planning [75]. Bruce and Veloso [188] reported that decoupling path planning, using execution extended RRT (ERRT) [72], and motion control, produced more accurate and reliable results, especially when fast computations are needed. Recently, successful implementations of kinodynamic SBP have been achieved by limiting the planning dimensions and planning in the task-space [76], [189] and using visual information for localization [190]. Similarly, RG-RRT implemented in task space with the use of motion primitives, fulfilled the task of real-time kinodynamic motion planning [191].

A promising approach is adopted by S-RRT. It encodes the constraints of an underactuated vehicle in the characteristics of a Bezier curve used for local planning [104]. Kinodynamic trajectory generation using B-spline [149] and Bezier curves [192] is widely studied and can be utilized by SBP to generate effective kinodynamic planners. The local modification support was exploited by generating a feasible path and then subsequent local adjustments are performed to ensure dynamic feasibility [193]. The main advantage, of using splines, is that kinodynamic planning is limited to a lower dimensional space, a notion similar to maneuver-based planning proposed by [137], thus planning can be executed in real time scenarios. Subsequently, B-spline interpolation was used to generate smooth trajectories for an RRT planner in a dynamic driving scenario [194].

V. OPTIMAL SBP

The ability of SBP to provide valid solutions for constrained high dimensional problems within a reasonable timeframe is advantageous. Despite the fact that the hit-or-miss sampling approach is the core of the planner's effective strategy, it leads to the inclusion of many redundant maneuvers in the final path. SBP generate highly suboptimal solutions and they are highly sensitive to their implementation details, as shown experimentally in section (III-J).

LaValle and Kuffner [60] proposed modification of the termination condition in a way such that the SBP keeps running to iteratively converge the path cost. The solution convergence remained an unanswered problem, until it was proven that given infinite runtime RRT will not find an optimal solution [195]. Numerous variants, such as k-RRT, Anytime RRT, and post processing methods have been proposed to remedy the poor solutions returned by RRT. Despite their effectiveness they provide no theoretical guarantees for reaching an optimal solution.

A. RRT*

A recent development in SBP was set forth by Karaman and Frazzoli [196]. A family of optimal sampling based planners, RRT*[2], PRM* and RRG*, were presented which guaranteed asymptotic optimality. These algorithms operate analogously to any common SBP except in two procedures. Performing nearest neighbor search and adding a node to the existing graph or tree. The two different procedures are named "Near vertices" and "Rewire". *Near vertices* returns a number of nearest nodes similar to k-RRT [121]. In the case of RRT*, the nodes are returned, if they are within a ball of radius, k. This ball radius is a function of the number of nodes in the tree, n, and is defined by Eqn. (4), where γ is a parameter based on the environment characteristics and d is the C-space dimension [196].

$$k = \gamma \left(\frac{\log (n)}{(n)} \right)^{1/d} \tag{4}$$

The nearest vertices are returned within a ball of radius k and stored in a set Q_{near}, as shown in Fig. 19(b). The selected node, $q_{new,}$ is connected to the node, q_{parent}, which provides a shorter router to the start configuration, as shown in Fig. 19(c). All remaining nodes in Q_{near} are rewired to q_{new} as their parent, if it provides a shorter route to the start configuration, as shown in Fig. 19(d). Hence every new node, q_{new}, will endeavor to improve all local connections within a predefined radius. An RRT* tree is shown in Fig. 18 after 6,000 iterations.

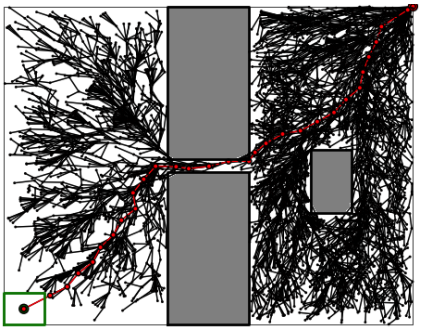

FIGURE 18. RRT* tree after 6,000 iterations and 4,700 explored nodes.

B. REQUIREMENTS FOR RRT* TO GENERATE OPTIMAL SOLUTIONS

The realization of an optimal solution dictates some criteria that must be met. Primarily, optimality is defined with respect to a specific metric and the planner is constantly attempting to enhance the value of that metric. As previously discussed SBP, defining a true metric that signifies the cost between two configurations has proved to be a non-trivial task.

In addition to defining a metric, a steering function must be defined in the planner. RRT* [196] relies on the existence of a steering function that drives the robot through an optimal trajectory between two specified states or configurations. A likewise assumed *guidance loop* is the core of the work by Frazzoli, et al. [137]. Such steering function does not exist for several robotic systems. Optimal control is still a subject pursued by researchers even for simple path planning purposes [197]. An alternative to defining a steering function is storing optimal trajectories and picking a suitable

[2]pronounced RRT star

trajectory when connecting two configurations, a particularly useful strategy for redundant articulated manipulators [74], or differentially constrained dynamic systems [170].

Even so for a holonomic system, whose optimal path is the straight line joining two configurations, the planner still guarantees only asymptotic optimality. This property indicates that the planner will always reach an optimal solution when the runtime approaches infinite. The initial solutions will be suboptimal, similar to RRT, and it will continue to converge towards optimality as the planner is running.

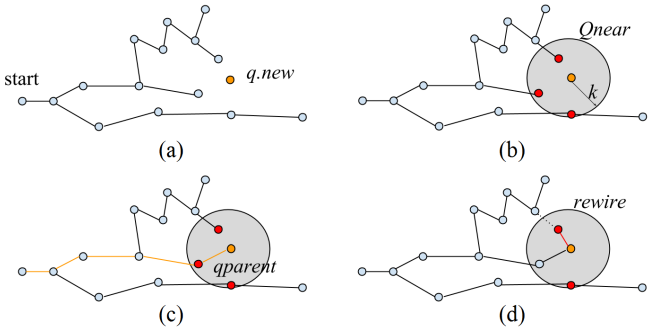

FIGURE 19. Illustrating the operation of RRT* (a) A new random node, q_{new}, is selected, shown as orange node (b) Near vertices procedure returns a set, Q_{near}, of all nodes, shown as red nodes, within a certain distance of the new node (circular area shaded in grey) (c) q_{new} is connected to the node, q_{parent}, that has the shortest route to the start (shown as orange path) (d) The remaining nodes in Q_{near} are rewired through q_{new}, if it provides a shorter path towards the start.

C. CONVERGENCE TOWARDS AN OPTIMAL SOLUTION

RRT* is guaranteed to asymptotically converge towards an optimal solution under certain assumptions. The convergence rate, however, has been shown to be rather slow. In fact, certain post processing approaches outperform RRT* [151]. The desirable properties of RRT* in real-time applications are overshadowed by the planning time wasted to reach an optimal solution.

There are a number of methods that endeavor to speed up the convergence of RRT*. A bidirectional RRT* that only joins promising nodes have been shown to improve the performance in high dimensional spaces [198]. The node selection strategy is similar to the one employed by Anytime RRT [138].

RRT*-smart removes redundant nodes every planning iteration and biases the sampling towards the remaining nodes [199]. A naive algorithm is implemented to trim the tree as it checks subsequent nodes. A possible improvement is the use path refinement algorithm presented in [140]. RRT*-smart resembles the anytime meta-algorithm presented in [151], as it alternates between post processing and expanding the tree.

A potential field function is coupled with the RRT* algorithm to guide the algorithm towards the optimal solution [200]. It attempts to strike a balance between exploitation and exploration as suggested by [102]. However, the presented approach does not adaptively change the behavior and the parameters are predetermined prior to planning.

NN search is identified as a bottleneck in SBP. RRT* proceeds to compute a set of near vertices, in each iteration, that lie within a ball of known radius. The convergence rate is accelerated by approximating costs between nodes, when computing nearest vertices for a certain node [201].

It can be observed that a large fraction of the RRT* planning time is spent extending the tree into areas that are not necessarily promising, adding and rewiring redundant nodes. This is illustrated in Fig. 18. All areas are heavily sampled even though most of those nodes will not contribute to the path optimality.

To overcome the slow convergence rate an Anytime framework for RRT* was implemented on an autonomous forklift [202]. Anytime RRT* finds a suboptimal path and converges towards optimality within the given planning time. This anytime implementation is suitable for real-time applications, as the planner must return a path whenever it is called i.e. existence of a path in real applications is far more vital than the path optimality.

Node selection criteria were impose to limit the addition of nodes whose shortest path is large than a certain bound [198], [203]. Modifying the rewiring procedure to include the nearest nodes in the shortest path increases the convergence rate [203]. A predictive model is used to estimate the probability of a node being on the optimal path and is used to guide the path towards optimal regions [105]. RRT#[3] replaces the local rewiring procedure by globally replanning the path [204]. Efficiently updating all the node costs, and categorizing nodes such that only promising nodes will be expanded, is the basis for this planner. In this context, promising nodes are those, which can constitute an optimal path. It is shown that RRT# converges faster towards an optimal solution as it guarantees an optimal solution is returned, given all the present node costs.

D. OPTIMAL KINODYNAMIC PLANNING

The development of optimal planning and RRT* algorithm has renewed interest in SBP. As an example, RRT* has been extended for vector fields, not just uniform environments [205]. It also led to the emergence of research in optimal kinodynamic planning Karaman and Frazzoli [196] argued that RRT* is analogous to RRT, thus is a generalized planner that can be applied in any planning context. Conceptually this statement is accurate, however, in a practical sense it is a perplexing task to apply optimal SBP in kinodynamic, real-time or, dynamic scenarios.

At this point, there are a handful of optimal kinodynamic planning planners. They are limited to systems with linear dynamics [206], [207], whose cost functions are well known and can be computed between any two states.

Optimal kinodynamic SBP for differentially constrained, high dimensional systems was achieved [208] by limited the planning to the task space [189] and using reachability guided trees [186]. Planning with task space is a general approach

[3]pronounced RRT sharp

that can be adapted to planning relative to the end-effector of a manipulator or a center of mass of a robot. A more challenging problem, non-holonomic kinodynamic SBP, was resolved similarly [209]. Nonetheless, these planners are still restricted to simulation-based applications due to their high computational requirements.

E. EXPERIMENTS AND RESULTS

A preliminary version of this analysis was presented in [203]. The aim of these experiments is twofold,

- Highlight the observation that RRT*, much like RRT, is sensitive to its implementation parameters which must be carefully chosen,
- Demonstrate the benefits of applying node selection criteria on RRT*.

1) PARAMETER SENSITIVITY

As expected, the implementation of RRT* is of grave importance to its performance. The results of modifying the step size can be seen in Fig. 20. Similar to RRT, decreasing the step size of the planner extension improve the overall path cost. To eliminate redundant nodes that will not contribute to the path convergence, a minimum step size has been specified.

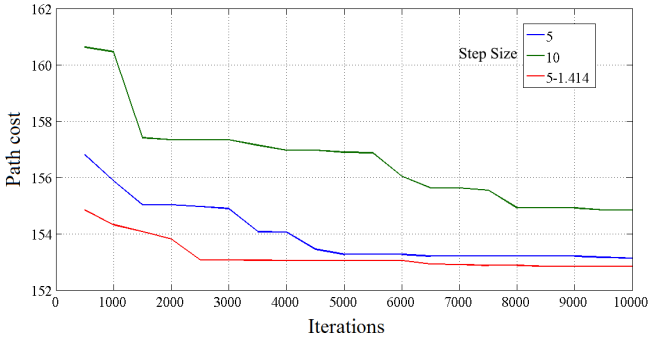

FIGURE 20. Effect of step size on the path cost and convergence rate.

Goal biasing is employed to speed up the performance of RRT and guide it towards finding a solution. It has successfully done so for RRT* as well, as can be seen in Fig. 21. The planner was unable to find a solution before 1,500 iterations. However with biasing it was successful before reaching 500 iterations. Additionally, biasing improves the cost of the initial solution found by RRT* and it decreases its convergence. Biasing is recommended prior to find a quick solution and then it has to be terminated due to its negative effect on the convergence rate. Biasing the RRT* can be an alternative to the recommended use of a traditional RRT to find an initial solution in [208] and [209].

2) NODE SELECTION

The node selection strategy, exploited by Anytime RRT to bound sampling merely to promising nodes has been proposed as a performance enhancement for RRT*. The lower bound is defined by, ε, where the lower bound equals (1- ε) times the current path cost. It is generally taken between

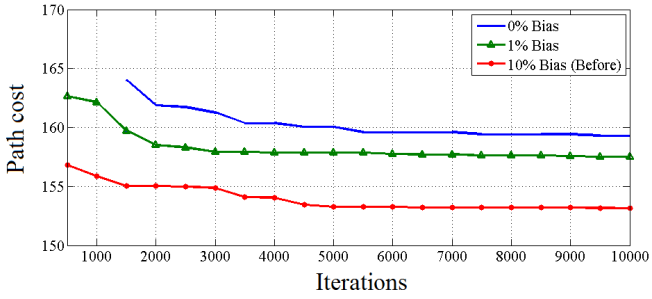

FIGURE 21. Effect of goal biasing on path cost and convergence rate.

5%-10% and indicates the improvement in the path cost. This leads to generating sparse trees, as shown in Table 4. The effect of node selection is illustrated by comparing Fig. 22 and Fig. 18. The planner generates the almost identical solutions with far less nodes explored. Aside from merely adding promising nodes that will lead to better solutions, the work by [186] argued for the effectiveness of maintaining sparse trees especially for non-holonomic kinodynamic planning.

TABLE 4. Average number of nodes in the tree after 10,000 iterations.

RRT*	$\varepsilon = 1\%$	$\varepsilon = 5\%$
7731	4718	4066

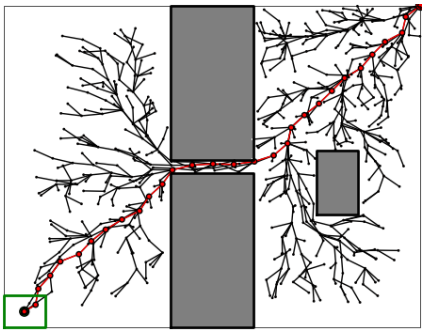

FIGURE 22. A sparse RRT* tree generated, after 6,000 iterations. Node selection has been employed with $\varepsilon = 5\%$.

VI. DYNAMIC AND UNCERTAIN ENVIRONMENTS

A common assumption in planning algorithms is that the environment is well defined such that the robot's location relative to obstacles and goal positioned are all known. This statement holds true in static environments where industrial manipulators are used or in CAD applications in which the environment is user-defined.

Autonomous vehicles and robots operate in dynamic changing environments with other uncontrollable, in some cases lethargic, agents that cannot be modeled or estimated. In general, the assumption of a well-defined static environment does not hold. There is an uncertainty that arises as a result of sensing errors and noise and the imprecision of actuators and other uncontrollable factors such as wheel slip.

Consequently, the exact location of the robot (localization) and the description of the environment (mapping) is not a trivial task. In this section we present planners that tackle one of the two current issues in robotic motion planning, replanning in dynamic and/or uncertain environments. Early on, it was assumed that, since SBP were able to generate relatively rapid solutions, it would suffice to discard current solutions and replan when deviations were identified in the environment.

Regenerating a single-query search tree may be a valid approach, given the appropriate parameters and heuristics in certain instances. In the case for multi-query planners, such as PRM, that invest most of their resources in connecting the environment, regenerating the entire roadmap is not feasible.

An outline for using PRM in dynamic environments involved generating a roadmap while assuming an obstacle free space [210]. The data structure of PRM was made more efficient in order to accommodate changes in the environment and consequently in the roadmap. A similar approach attempts to use single query planner to connect PRM nodes in dynamic environments and encodes obstacle positions in local connections [211]. van den Berg, et al. [212] proposed a generalized PRM method in surroundings where obstacle movements are restricted to local sectors. PDR maintains a roadmap whose paths can be deformed, thus numerous paths can be obtained between two configurations [152]. PDR has been proposed for dynamic path planning by the authors, but is yet to be evaluated in those scenarios.

RRF (Reconfigurable Random Forests) provided a framework to managing either roadmap, or tree planners, under changing settings [213]. Once changes in the environment are detected, nodes in C_{obs} and colliding paths are discarded. This leads to the emergence of separated roadmaps, or forests. The planner then prunes the forests and attempts reconnects paths. Lazy reconfiguration forest (LRF) used the same framework but proceeded to perform collision checking only for the paths involved for planning [214].

ERRT is often mentioned as the first algorithm to be implemented in a real-time dynamic situation [72], [188]. ERRT maintains a single tree. If that tree is collides with the obstacle space it is discarded and another one is rebuilt. ERRT maintains the location of the discarded configurations, waypoint cache, and biasing the search slightly towards those node locations. It is motivated by the assumption that, if the algorithm is updated at a high frequency, a small percentage of the original tree needs to be modified.

Dynamic RRT (DRRT) builds on the idea that it is more efficient to repair the existing tree, than to, rebuild an entirely new one [215]. Unlike ERRT, only the colliding configurations and their child nodes are discarded in an efficient manner. DRRT borrows the concept of slightly biasing the search towards invalidated areas from ERRT. Nonetheless, it outperforms ERRT by repairing the tree. AD* was coupled with PRM to provide an efficient framework for replanning [216]. In this approach the motion of other agents was extrapolated, the planner failed to generate solutions when worst-case scenario of a growing disc is considered. Growing discs assumption creates a narrow free regions in the C-space. A scenario in which SBP perform poorly. Different types of dynamic obstacles are shown in Fig. 23 and worst-case growing discs are shown on the far right. Flexible-PRM (F-PRM) similarly used backward A*, from the static goal towards the moving robot, in dynamic environments [124].

Multipartite RRT (MP-RRT) [217] combines the strategy of biasing the search towards discarded configurations, similar to ERRT. It also rebuilds the tree, like DRRT, and maintains separate detached forests, like RRF. MP-RRT distinguishes itself from RRF by only maintaining forests for a limited time so as not to waste computational time in unpromising areas.

In the course of navigating a dynamic environment, it is possible that a plan is deemed unsafe, such that it will collide with a moving obstacle. The selected planning framework must generate an alternate feasible route. The concept τ-safety ensures that at any stage during path execution there is enough time, τ, for the planner to compute an alternate path while following the unsafe path [137]. Greedy, Incremental, Path-Directed (GRIP) [180] is a safe, replanning framework that guarantees safety by considering Inevitable collisions states (ICS) during replanning and only considering safety-guarantees to reduce planning time. GRIP employs a similar rebuilding strategy to ERRT and DRRT, node selection is based on coverage, as employed by PDST, and an efficient safe framework based on τ-safety.

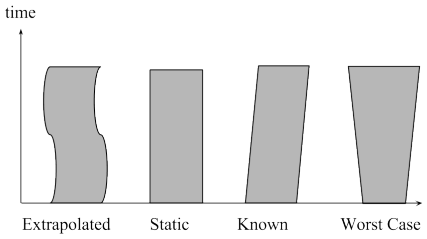

FIGURE 23. Dynamic obstacles different trajectory assumptions with respect to time.

Considering stochastic sensing and dynamic conditions is a relatively novel topic in motion planning. Particle RRT [218] and RRT-SLAM [219] model uncertainty using particle filters, which is then considered in the planning. Similar uncertainty considerations are added to RRT* framework by Rapidly-exploring Random Belief Trees (RRBT) [220], [221]. To guarantee the accuracy of planned path, uncertainty is encoded in path costs to guide the robot to useful areas and thus ensuring the robot will not be lost without information. Gaussian processes were also used to predict the motion of other vehicles in the environment [222]. The planner estimates the probability of collision and returns a path that is probabilistically collision free. This approach may serve as an alternative to worst-case growing discs model, however, the objects in the environment must be analyzed prior to planning. EG-RRT [123] evaluates the collision probability

of each state in RRT tree based on the modeled uncertainty of dynamics and sensing and creates a cost map of the environment.

TABLE 5. Summary of replanning strategies.

Planner	Dynamic	Uncertain	Safe
PRM *(dynamic)* [210-212]	X	-	-
RRF [213]	X	-	-
LRF [214]	X	-	-
ERRT [72, 188]	X	-	-
DRRT [215]	X	-	-
van den Berg, et al. [216]	X	-	-
F-PRM [124]	X	-	-
MP-RRT [217]	X	-	-
Frazzoli, et al. [137]	X	-	X
GRIP [180]	X	-	X
Probabilistic RRT [222]	X	X	-
RRBT [220, 221]	-	X	X
Particle RRT [218]	-	X	-
RRT-SLAM [219]	-	X	-
EG-RRT [123]	-	X	-
GRRT and GPRM [223]	-	X	-
FIRM [224]	-	X	-

Generalized RRT and PRM have been proposed in which the robot dynamics and sensors are stochastically modeled and the local planner estimates the probability of transition success and will not proceed if the probability exceeds a threshold [223]. Unlike traditional planners, generalized planners terminate only when a solution with a high probability of success is found. Feedback-based Information Roadmap (FIRM) also relies on feedback from local planners to reduce the uncertainty propagation between states [224]. The question of path planning amongst moving uncontrollable obstacles, under stochastic dynamic and sensing conditions represents a next step in robotic research. The amalgamation of uncertainty, kinodynamic, and optimal planning in active environments is bound to push robots into new frontiers. Planning strategies in this section are categorized into dynamic, uncertain and safe in Table 5.

VII. CONCLUSION

Sampling-based planning has established its success in solving the intricate problem of robot motion planning. Numerous methods have been proposed to improve the efficiency of planning and the quality of plans. They have provided significant advantages in a wide field of real-life and simulation based scenarios. Subsequently, there exists an immense body of work on the topic of SBP.

This brings into the light the gap that existed in classical motion planning algorithms to illustrate the significance of SBP and the motivation behind their inception. We then continue by listing the main algorithms that all state of the art algorithms are built upon. The growing body of work is surveyed in this paper. Several experiments are executed to shed a light on some of the current issues, investigated by researchers, and highlight the implementation details that are often not discussed when planners and algorithms are proposed.

An emphasis is placed on the contemporary research problems. Motion planning has moved away from path planning in static environments for simple robots into more challenging arenas. Methods that address real-time kinodynamic planning, optimal planning, planning under uncertainty and in dynamic environments are given particular attention. The continued expansion of robot motion planning boundaries promises more realistic methods that can be utilized in real-life scenarios.

REFERENCES

[1] A. Kelly and A. Stentz, "Rough terrain autonomous mobility—Part 2: An active vision, predictive control approach," *Autonom. Robot.*, vol. 5, no. 2, pp. 163–198, May 1998.
[2] P. Corke, J. Lobo, and J. Dias, "An introduction to inertial and visual sensing," *Int. J. Robot. Res.*, vol. 26, no. 6, pp. 519–535, Jun. 2007.
[3] G. Atanacio-Jiménez, J. J. González-Barbosa, J. B. Hurtado-Ramos, F. J. Ornelas-Rodríguez, H. Jiménez-Hernández, T. García-Ramirez, *et al.*, "LIDAR velodyne HDL-64E calibration using pattern planes," *Int. J. Adv. Robot. Syst.*, vol. 8, no. 5, pp. 70–82, Oct. 2011.
[4] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford campus vision and lidar data set," *Int. J. Robot. Res.*, vol. 30, no. 13, pp. 1543–1552, Nov. 2011.
[5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.
[6] J. Borenstein and F. Liqiang, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, pp. 869–880, Dec. 1996.
[7] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006.
[8] D. Ferguson, C. Baker, M. Likhachev, and J. Dolan, "A reasoning framework for autonomous urban driving," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2008, pp. 775–780.
[9] J. G. Hurdus and D. W. Hong, "Behavioral programming with hierarchy and parallelism in the DARPA urban challenge and robocup," in *Proc. IEEE Int. Conf. MFI Intell. Syst.*, Aug. 2008, pp. 503–509.
[10] R. Wallace, A. Stentz, C. Thorpe, H. Moravec, W. Whittaker, and T. Kanade, "First results in robot road following," in *Proc. IJCAI*, 1985, pp. 381–387.
[11] G. Antonelli, S. Chiaverini, and G. Fusco, "A fuzzy-logic-based approach for mobile robot path tracking," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 2, pp. 211–221, Apr. 2007.
[12] R. N. Jazar, "Mathematical theory of autodriver for autonomous vehicles," *J. Vibrat. Control*, vol. 16, no. 2, pp. 253–279, Feb. 2010.
[13] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, May 2006.
[14] J.-C. Latombe, *Robot Motion Planning*. New York, NY, USA: Springer-Verlag, 1990.
[15] H. M. Choset, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2005.
[16] J.-C. Latombe, "Motion planning: A journey of robots, molecules, digital actors, and other artifacts," *Int. J. Robot. Res.*, vol. 18, no. 11, pp. 1119–1128, Nov. 1999.
[17] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proc. 20th Annu. Symp. Found. Comput. Sci.*, Oct. 1979, pp. 421–427.
[18] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility-polygon search and euclidean shortest paths," in *Proc. 26th Annu. Symp. Found. Comput. Sci.*, Oct. 1985, pp. 155–164.
[19] C. Alexopoulos and P. M. Griffin, "Path planning for a mobile robot," *IEEE Trans. Syst., Man Cybern.*, vol. 22, no. 2, pp. 318–322, Mar./Apr. 1992.
[20] T. Maekawa, T. Noda, S. Tamura, T. Ozaki, and K.-I. Machida, "Curvature continuous path generation for autonomous vehicle using B-spline curves," *Comput.-Aided Des.*, vol. 42, no. 4, pp. 350–359, Apr. 2010.

[21] J. Canny, "A Voronoi method for the piano-movers problem," in *Proc. IEEE Int. Conf. Robot. Autom.*, Mar. 1985, pp. 530–535.

[22] O. Takahashi and R. J. Schilling, "Motion planning in a plane using generalized Voronoi diagrams," *IEEE Trans. Robot. Autom.*, vol. 5, no. 2, pp. 143–150, Apr. 1989.

[23] G. E. Jan, C. C. Sun, W. C. Tsai, and T. H. Lin, "An $O(nlogn)$ shortest path algorithm based on delaunay triangulation," *IEEE/ASME Trans. Mechatron.*, Feb. 2013.

[24] M. Elbanhawi, M. Simic, and R. Jazar, "Autonomous mobile robot path planning: A novel roadmap approach," *Appl. Mech. Mater.*, vols. 373–375, pp. 246–254, Jan. 2013.

[25] R. A. Brooks and T. Lozano-Perez, "A subdivision algorithm in configuration space for findpath with rotation," *IEEE Trans. Syst., Man Cybern.*, vol. 15, no. 2, pp. 224–233, Mar./Apr. 1985.

[26] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[27] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[28] A. Stentz, "Optimal and efficient path planning for unknown and dynamic environments," *Int. J. Robot. Autom.*, vol. 10, no. 3, pp. 89–100, 1995.

[29] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime search in dynamic graphs," *Artif. Intell.*, vol. 172, no. 14, pp. 1613–1643, Sep. 2008.

[30] T. Howard, C. Green, and A. Kelly, "State space sampling of feasible motions for high performance mobile robot navigation in highly constrained environments," in *Field and Service Robotics*, vol. 42, C. Laugier and R. Siegwart, Eds. Berlin, Germany: Springer-Verlag, 2008, pp. 585–593.

[31] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *J. Field Robot.*, vol. 26, no. 3, pp. 308–333, Mar. 2009.

[32] M. Pivtoraiko and A. Kelly, "Kinodynamic motion planning with state lattice motion primitives," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2011, pp. 2172–2179.

[33] H. R. Beom and H. S. Cho, "A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning," *IEEE Trans. Syst., Man Cybern.*, vol. 25, no. 3, pp. 464–477, Mar. 1995.

[34] H. Martínez-Alfaro and S. Gómez-García, "Mobile robot path planning and tracking using simulated annealing and fuzzy logic control," *Expert Syst. Appl.*, vol. 15, nos. 3–4, pp. 421–429, Oct./Nov. 1998.

[35] H. Seraji and A. Howard, "Behavior-based robot navigation on challenging terrain: A fuzzy logic approach," *IEEE Trans. Robot. Autom.*, vol. 18, no. 3, pp. 308–321, Jun. 2002.

[36] D. Janglová, "Neural networks in mobile robot motion," *Int. J. Adv. Robot. Syst.*, vol. 1, no. 1, pp. 15–22, 2004.

[37] M. Gerke, "Genetic path planning for mobile robots," in *Proc. Amer. Control Conf.*, vol. 4. Jun. 1999, pp. 2424–2429.

[38] H. Yanrong, S. X. Yang, X. Li-zhong, and M. Q. H. Meng, "A knowledge based genetic algorithm for path planning in unstructured mobile robot environments," in *Proc. IEEE Int. Conf. Robot. Biomimet.*, Aug. 2004, pp. 767–772.

[39] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda, and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Appl. Soft Comput.*, vol. 9, no. 3, pp. 1102–1110, Jun. 2009.

[40] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, Mar. 1986.

[41] R. C. Arkin, "Motor schema—Based mobile robot navigation," *Int. J. Robot. Res.*, vol. 8, no. 4, pp. 92–112, Aug. 1989.

[42] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2. Apr. 1991, pp. 1398–1404.

[43] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE Trans. Robot. Autom.*, vol. 7, no. 3, pp. 278–288, Jun. 1991.

[44] R. Simmons, "The curvature-velocity method for local obstacle avoidance," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 4. Apr. 1996, pp. 3375–3382.

[45] F. Belkhouche and B. Bendjilali, "Reactive path planning for 3-D autonomous vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 1, pp. 249–256, Jan. 2012.

[46] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *Int. J. Robot. Res.*, vol. 26, no. 2, pp. 141–166, Feb. 2007.

[47] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *Int. J. Robot. Res.*, vol. 31, no. 3, pp. 346–359, Mar. 2012.

[48] B. R. Donald, "A search algorithm for motion planning with six degrees of freedom," *Artif. Intell.*, vol. 31, no. 3, pp. 295–353, Mar. 1987.

[49] N. Metropolis and S. Ulam, "The Monte Carlo method," *J. Amer. Statist. Assoc.*, vol. 44, no. 247, pp. 335–341, Sep. 1949.

[50] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953.

[51] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, no. 6, pp. 628–649, Dec. 1991.

[52] S. Carpin and G. Pillonetto, "Motion planning using adaptive random walks," *IEEE Trans. Robot.*, vol. 21, no. 1, pp. 129–136, Feb. 2005.

[53] L. Kavraki and J. C. Latombe, "Randomized preprocessing of configuration space for path planning: Articulated robots," in *Proc. IEEE/RSJ/GI Int. Conf. IROS*, vol. 3. Sep. 1994, pp. 1764–1771.

[54] N. M. Amato and Y. Wu, "A randomized roadmap method for path and manipulation planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1. Apr. 1996, pp. 113–120.

[55] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[56] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dept. Comput. Sci., Iowa State Univ., Ames, IA, USA, Tech. Rep. TR 98-11, 1998.

[57] P. Švestka and M. H. Overmars, "Motion planning for carlike robots using a probabilistic learning approach," *Int. J. Robot. Res.*, vol. 16, no. 2, pp. 119–143, Apr. 1997.

[58] J. Barraquand, L. Kavraki, J.-C. Latombe, R. Motwani, T.-Y. Li, and P. Raghavan, "A random sampling scheme for path planning," *Int. J. Robot. Res.*, vol. 16, no. 6, pp. 759–774, Dec. 1997.

[59] D. Hsu, J.-C. Latombe, and H. Kurniawati, "On the probabilistic foundations of probabilistic roadmap planning," *Int. J. Robot. Res.*, vol. 25, no. 7, pp. 627–643, Jul. 2006.

[60] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001.

[61] D. Hsu, J. C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3. Apr. 1997, pp. 2719–2726.

[62] J. M. Ahuactzin, K. Gupta, and E. Mazer, "Manipulation planning for redundant robots: A practical approach," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 731–747, Jul. 1998.

[63] J. Kuffner, "Autonomous agents for real-time animation," Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 1999.

[64] I. Karamouzas and M. Overmars, "Simulating and evaluating the local behavior of small pedestrian groups," *IEEE Trans. Visualizat. Comput. Graph.*, vol. 18, no. 3, pp. 394–406, Mar. 2012.

[65] Y. Kwangjin, S. K. Gan, and S. Sukkarieh, "A Gaussian process-based RRT planner for the exploration of an unknown and cluttered environment with a UAV," *Adv. Robot.*, vol. 27, no. 6, pp. 431–443, Apr. 2013.

[66] I. Al-Bluwi, T. Siméon, and J. Cortés, "Motion planning algorithms for molecular simulations: A survey," *Comput. Sci. Rev.*, vol. 6, no. 4, pp. 125–143, Jul. 2012.

[67] B. Gipson, D. Hsu, L. E. Kavraki, and J.-C. Latombe, "Computational models of protein kinematics and dynamics: Beyond simulation," *Annu. Rev. Anal. Chem.*, vol. 5, pp. 273–291, Jul. 2012.

[68] C. S. Han, K. H. Law, J.-C. Latombe, and J. C. Kunz, "A performance-based approach to wheelchair accessible route analysis," *Adv. Eng. Informat.*, vol. 16, no. 1, pp. 53–71, 2002.

[69] R. Alterovitz, M. Branicky, and K. Goldberg, "Motion planning under uncertainty for image-guided medical needle steering," *Int. J. Robot. Res.*, vol. 27, nos. 11–12, pp. 1361–1374, Nov. 2008.

[70] M. Moll and L. E. Kavraki, "Path planning for deformable linear objects," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 625–636, Aug. 2006.

[71] M. S. Branicky, M. M. Curtiss, J. A. Levine, and S. B. Morgan, "RRTs for nonlinear, discrete, and hybrid planning and control," in *Proc. 42nd IEEE Conf. Decision Control*, vol. 1. Dec. 2003, pp. 657–663.

[72] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 3. Oct. 2002, pp. 2383–2388.
[73] A. L. Olsen and H. G. Petersen, "Motion planning for gantry mounted manipulators: A ship-welding application example," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 4782–4786.
[74] L.-P. Ellekilde and H. G. Petersen, "Motion planning efficient trajectories for industrial bin-picking," *Int. J. Robot. Res.*, vol. 32, pp. 991–1004, Aug. 2013.
[75] S. Srinivasa, D. Ferguson, C. Helfrich, D. Berenson, A. Collet, R. Diankov, *et al.*, "HERB: A home exploring robotic butler," *Auto. Robots*, vol. 28, no. 1, pp. 5–20, Jan. 2010.
[76] S. S. Srinivasa, D. Berenson, M. Cakmak, A. Collet, M. R. Dogar, A. D. Dragan, *et al.*, "HERB 2.0: Lessons learned from developing a mobile manipulator for the home," *Proc. IEEE*, vol. 100, no. 8, pp. 2410–2428, Jul. 2012.
[77] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. P. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, Sep. 2009.
[78] Y. Jihyun and C. D. Crane, "Path planning for Unmanned Ground Vehicle in urban parking area," in *Proc. 11th ICCAS*, Oct. 2011, pp. 887–892.
[79] S. Lindemann and S. LaValle, "Current issues in sampling-based motion planning," in *Robotics Research*, vol. 15, P. Dario and R. Chatila, Eds. Berlin, Germany: Springer-Verlag, 2005, pp. 36–54.
[80] K. I. Tsianos, I. A. Sucan, and L. E. Kavraki, "Sampling-based robot motion planning: Towards realistic applications," *Comput. Sci. Rev.*, vol. 1, no. 1, pp. 2–11, Aug. 2007.
[81] I. A. Sucan and L. E. Kavraki, "On the implementation of single-query sampling-based motion planners," in *Proc. IEEE ICRA*, Jan. 2010, pp. 2005–2011.
[82] I. A. Sucan, M. Moll, and E. E. Kavraki, "The open motion planning library," *IEEE Robot. Autom. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012.
[83] S. M. LaValle, "Motion planning part II: Wild frontiers," *IEEE Robot. Autom. Mag.*, vol. 18, no. 2, pp. 108–118, Feb. 2011.
[84] S. M. LaValle, "Motion planning part I: The essentials," *IEEE Robot. Autom. Mag.*, vol. 18, no. 1, pp. 79–89, Jan. 2011.
[85] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Trans. Comput.*, vol. 32, no. 2, pp. 108–120, Feb. 1983.
[86] R. Geraerts and M. H. Overmars, "Sampling and node adding in probabilistic roadmap planners," *Robot. Auto. Syst.*, vol. 54, no. 2, pp. 165–173, Feb. 2006.
[87] R. Diankov and J. Kuffner, "Randomized statistical path planning," in *Proc. IEEE/RSJ Int. Conf. IROS*, Nov. 2007, pp. 1–6.
[88] L. J. Guibas, C. Holleman, and L. E. Kavraki, "A probabilistic roadmap planner for flexible objects with a workspace medial-axis-based sampling approach," in *Proc. IEEE/RSJ Int. Conf. IROS*, vol. 1. Oct. 1999, pp. 254–259.
[89] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the space," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2. May 1999, pp. 1024–1031.
[90] C. Holleman and E. E. Kavraki, "A framework for using the workspace medial axis in PRM planners," in *Proc. IEEE Int. Conf. ICRA*, vol. 2. Apr. 2000, pp. 1408–1413.
[91] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2. May 1999, pp. 1018–1023.
[92] S. Zheng, D. Hsu, J. Tingting, H. Kurniawati, and J. H. Reif, "Narrow passage sampling for probabilistic roadmap planning," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1105–1115, Dec. 2005.
[93] D. Hsu and S. Zheng, "Adaptively combining multiple sampling strategies for probabilistic roadmap planning," in *Proc. IEEE Conf. Robot., Autom. Mechatron.*, vol. 2. Dec. 2004, pp. 774–779.
[94] S. Thomas, M. Morales, T. Xinyu, and N. M. Amato, "Biasing samplers to improve motion planning performance," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 1625–1630.
[95] T. Siméon, J. P. Laumond, and C. Nissoux, "Visibility-based probabilistic roadmaps for motion planning," *Adv. Robot.*, vol. 14, no. 6, pp. 477–493, Jan. 2000.
[96] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE ICRA*, vol. 2. Apr. 2000, pp. 995–1001.
[97] R. Geraerts and M. H. Overmars, "Reachability-based analysis for probabilistic roadmap planners," *Robot. Auto. Syst.*, vol. 55, no. 11, pp. 824–836, Nov. 2007.
[98] A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle, "Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain," in *Proc. IEEE ICRA*, Apr. 2005, pp. 3856–3861.
[99] L. Jaillet, A. Yershova, S. M. La Valle, and T. Simeon, "Adaptive tuning of the sampling domain for dynamic-domain RRTs," in *Proc. EEE/RSJ Int. Conf. IROS*, Aug. 2005, pp. 2851–2856.
[100] B. Burns and O. Brock, "Single-query motion planning with utility-guided random trees," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 3307–3312.
[101] S. Rodriguez, T. Xinyu, L. Jyh-Ming, and N. M. Amato, "An obstacle-based rapidly-exploring random tree," in *Proc. IEEE ICRA*, May 2006, pp. 895–900.
[102] M. Rickert, O. Brock, and A. Knoll, "Balancing exploration and exploitation in motion planning," in *Proc. IEEE ICRA*, May 2008, pp. 2812–2817.
[103] O. Brock and E. E. Kavraki, "Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces," in *Proc. IEEE ICRA*, vol. 2. May 2001, pp. 1469–1474.
[104] Y. Kwangjin, "An efficient Spline-based RRT path planner for non-holonomic robots in cluttered environments," in *Proc. ICUAS*, May 2013, pp. 288–297.
[105] M. Kobilarov, "Cross-entropy motion planning," *Int. J. Robot. Res.*, vol. 31, no. 7, pp. 855–871, Jun. 2012.
[106] R. A. Knepper and M. T. Mason, "Real-time informed path sampling for motion planning search," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1231–1250, Sep. 2012.
[107] R. Wein, J. van den Berg, and D. Halperin, "Planning high-quality paths and corridors amidst obstacles," *Int. J. Robot. Res.*, vol. 27, nos. 11–12, pp. 1213–1231, Nov./Dec. 2008.
[108] J. M. Phillips, N. Bedrossian, and E. E. Kavraki, "Guided expansive spaces trees: A search strategy for motion- and cost-constrained state spaces," in *Proc. IEEE ICRA*, vol. 4. Apr./May 2004, pp. 3968–3973.
[109] A. Ladd and L. Kavraki, "Fast tree-based exploration of state space for robots with dynamics," in *Algorithmic Foundations of Robotics VI*, vol. 17, M. Erdmann, M. Overmars, D. Hsu, and F. der Stappen, Eds. Berlin, Germany: Springer-Verlag, 2005, pp. 297–312.
[110] I. Sucan and L. E. Kavraki, "A sampling-based tree planner for systems with complex dynamics," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 116–131, Feb. 2012.
[111] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "Choosing good distance metrics and local planners for probabilistic roadmap methods," *IEEE Trans. Robot. Autom.*, vol. 16, no. 4, pp. 442–447, Aug. 2000.
[112] J. J. Kuffner, "Effective sampling and distance metrics for 3D rigid body path planning," in *Proc. IEEE ICRA*, vol. 4. Apr./May 2004, pp. 3993–3998.
[113] H. Long, D. Q. Huy, and S. Mita, "Unified path planner for parking an autonomous vehicle based on RRT," in *Proc. IEEE ICRA*, May 2011, pp. 5622–5627.
[114] P. Leven and S. Hutchinson, "Using manipulability to bias sampling during the construction of probabilistic roadmaps," *IEEE Trans. Robot. Autom.*, vol. 19, no. 6, pp. 1020–1026, Dec. 2003.
[115] C. Peng and S. M. LaValle, "Reducing metric sensitivity in randomized trajectory design," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1. Nov. 2001, pp. 43–48.
[116] C. Peng and S. M. LaValle, "Resolution complete rapidly-exploring random trees," in *Proc. IEEE ICRA*, vol. 1. Jan. 2002, pp. 267–272.
[117] J. Kim, J. M. Esposito, and V. Kumar, "Sampling-based algorithm for testing and validating robot controllers," *Int. J. Robot. Res.*, vol. 25, no. 12, pp. 1257–1272, Dec. 2006.
[118] M. Kalisiak and M. van de Panne, "RRT-blossom: RRT with a local flood-fill behavior," in *Proc. IEEE ICRA*, May 2006, pp. 1237–1242.
[119] S. Morgan and M. S. Branicky, "Sampling-based planning for discrete spaces," in *Proc. IEEE/RSJ Int. Conf. IROS*, vol. 2. Sep./Oct. 2004, pp. 1938–1945.
[120] M. S. Branicky, M. M. Curtiss, J. Levine, and S. Morgan, "Sampling-based planning, control and verification of hybrid systems," *IEE Proc. Control Theory Appl.*, vol. 153, no. 5, pp. 575–590, Sep. 2006.
[121] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," in *Proc. IEEE/RSJ Int. Conf. IROS*, vol. 2. Oct. 2003, pp. 1178–1183.

[122] M. B. Kobilarov and G. Sukhatme, "Near time-optimal constrained trajectory planning on outdoor terrain," in *Proc. IEEE ICRA*, Apr. 2005, pp. 1821–1828.

[123] L. Jaillet, J. Hoffman, J. van den Berg, P. Abbeel, J. M. Porta, and K. Goldberg, "EG-RRT: Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2011, pp. 2646–2652.

[124] K. Belghith, F. Kabanza, and L. Hartman, "Randomized path planning with preferences in highly complex dynamic environments," *Robotica*, vol. 31, no. 8, pp. 1195–1208, Dec. 2013.

[125] L. Jaillet, J. Cortes, and T. Simeon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robot.*, vol. 26, no. 4, pp. 635–646, Aug. 2010.

[126] J. Denny and N. M. Amato, "The toggle local planner for sampling-based motion planning," in *Proc. IEEE ICRA*, May 2012, pp. 1779–1786.

[127] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Tech.*, 1996, pp. 171–180.

[128] M. Reggiani, M. Mazzoli, and S. Caselli, "An experimental evaluation of collision detection packages for robot motion planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 3. 2002, pp. 2329–2334.

[129] C. L. Nielsen and E. E. Kavraki, "A two level fuzzy PRM for manipulation planning," in *Proc. IEEE/RSJ Int. Conf. IROS*, vol. 3. Nov. 2000, pp. 1716–1721.

[130] R. Bohlin and E. E. Kavraki, "Path planning using lazy PRM," in *Proc. IEEE ICRA*, vol. 1. Apr. 2000, pp. 521–528.

[131] J. Denny, K. Shi, and N. M. Amato, "Lazy toggle PRM: A single-query approach to motion planning," in *Proc. IEEE ICRA*, Apr. 2013, pp. 2407–2414.

[132] J. Bialkowski, S. Karaman, M. Otte, and E. Frazzoli, "Efficient collision checking in sampling-based motion planning," in *Algorithmic Foundations of Robotics X*, vol. 86, E. Frazzoli, T. Lozano-Perez, N. Roy, and D. Rus, Eds. Berlin, Germany: Springer-Verlag, 2013, pp. 365–380.

[133] G. Sánchez and J.-C. Latombe, "On delaying collision checking in PRM planning: Application to multi-robot coordination," *Int. J. Robot. Res.*, vol. 21, no. 1, pp. 5–26, Jan. 2002.

[134] W. Wang, X. Xu, Y. Li, J. Song, and H. He, "Triple RRTs: An effective method for path planning in narrow passages," *Adv. Robot.*, vol. 24, no. 7, pp. 943–962, Jan. 2010.

[135] W. Wang, Y. Li, X. Xu, and S. X. Yang, "An adaptive roadmap guided Multi-RRTs strategy for single query path planning," in *Proc. IEEE ICRA*, May 2010, pp. 2871–2876.

[136] C. Peng, E. Frazzoli, and S. Lavalle, "Improving the performance of sampling-based motion planning with symmetry-based gap reduction," *IEEE Trans. Robot.*, vol. 24, no. 2, pp. 488–494, Apr. 2008.

[137] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *J. Guid., Control, Dyn.*, vol. 25, no. 1, pp. 116–129, Jan. 2002.

[138] D. Ferguson and A. Stentz, "Anytime RRTs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 5369–5375.

[139] Z. Qi dan, W. Ye bin, W. Guo qiang, and W. Xin, "An improved anytime RRTs algorithm," in *Proc. Int. Conf. AICI*, Nov. 2009, pp. 268–272.

[140] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *Int. J. Robot. Res.*, vol. 26, no. 8, pp. 845–863, Aug. 2007.

[141] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, *et al.*, "Stanley: The robot that won the DARPA grand challenge," in *DARPA Grand Challenge*, vol. 36, M. Buehler, K. Iagnemma, and S. Singh, Eds. Berlin, Germany: Springer-Verlag, 2007, pp. 1–43.

[142] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," in *DARPA Urban Challenge*, vol. 56, M. Buehler, K. Iagnemma, and S. Singh, Eds. Berlin, Germany: Springer-Verlag, 2009, pp. 1–59.

[143] E. Papadopoulos, I. Poulakakis, and I. Papadimitriou, "On path planning and obstacle avoidance for nonholonomic platforms with manipulators: A polynomial approach," *Int. J. Robot. Res.*, vol. 21, no. 4, pp. 367–383, Apr. 2002.

[144] Y. Kwangjin and S. Sukkarieh, "An analytical continuous-curvature path-smoothing algorithm," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 561–568, Jun. 2010.

[145] Y. Kwangjin and S. Sukkarieh, "3D smooth path planning for a UAV in cluttered natural environments," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2008, pp. 794–800.

[146] K. G. Jolly, R. S. Kumar, and R. Vijayakumar, "A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits," *Robot. Auto. Syst.*, vol. 57, no. 1, pp. 23–33, Jan. 2009.

[147] Y. Kwangjin, D. Jung, and S. Sukkarieh, "Continuous curvature path-smoothing algorithm using cubic Bezier spiral curves for non-holonomic robots," *Adv. Robot.*, vol. 27, no. 4, pp. 247–258, Mar. 2013.

[148] Y. J. Kanayama and B. I. Hartman, "Smooth local-path planning for autonomous vehicles," *Int. J. Robot. Res.*, vol. 16, no. 3, pp. 263–284, Jun. 1997.

[149] M. Elbanhawi and M. Simic, "Spline framework for autonomous vehicles navigation," *Robotica, Under Rev.*, 2013, to be published.

[150] B. Raveh, A. Enosh, and D. Halperin, "A little more, a lot better: Improving path quality by a path-merging algorithm," *IEEE Trans. Robot.*, vol. 27, no. 2, pp. 365–371, Apr. 2011.

[151] R. Luna, I. A. Sucan, M. Moll, and L. E. Kavraki, "Anytime solution optimization for sampling-based motion planning," in *Proc. IEEE ICRA*, Oct. 2013, pp. 5068–5074.

[152] L. Jaillet and T. Simeon, "Path deformation roadmaps: Compact graphs with useful cycles for motion planning," *Int. J. Robot. Res.*, vol. 27, nos. 11–12, pp. 1175–1188, 2008.

[153] J. P. Laumond, S. Sekhavat, and F. Lamiraux, "Guidelines in nonholonomic motion planning for mobile robots," in *Robot Motion Planning and Control*, vol. 229, J. P. Laumond, Ed. Berlin, Germany: Springer-Verlag, 1998, pp. 1–53.

[154] B. Xuan-Nam, J.-D. Boissonnat, P. Soueres, and J. P. Laumond, "Shortest path synthesis for Dubins non-holonomic robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1. May 1994, pp. 2–7.

[155] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forward and backward," *Pacific J. Math.*, vol. 145, no. 2, pp. 367–393, Oct. 1990.

[156] T. Fraichard and A. Scheuer, "From Reeds and Shepp's to continuous-curvature paths," *IEEE Trans. Robot.*, vol. 20, no. 6, pp. 1025–1035, Dec. 2004.

[157] L. Z. Wang, K. T. Miura, E. Nakamae, T. Yamamoto, and T. J. Wang, "An approximation approach of the clothoid curve defined in the interval [0, Π/2] and its offset by free-form curves," *Comput.-Aided Des.*, vol. 33, no. 14, pp. 1049–1058, Dec. 2001.

[158] D. J. Walton, D. S. Meek, and J. M. Ali, "Planar G^2 transition curves composed of cubic Bézier spiral segments," *J. Comput. Appl. Math.*, vol. 157, no. 2, pp. 453–476, Aug. 2003.

[159] D. S. Meek and D. J. Walton, "An arc spline approximation to a clothoid," *J. Comput. Appl. Math.*, vol. 170, no. 1, pp. 59–77, Sep. 2004.

[160] D. J. Walton and D. S. Meek, "A controlled clothoid spline," *Comput. Graph.*, vol. 29, no. 3, pp. 353–363, Jun. 2005.

[161] J. McCrae and K. Singh, "Sketching piecewise clothoid curves," *Comput. Graph.*, vol. 33, no. 4, pp. 452–461, Aug. 2009.

[162] H. Yifeng and K. Gupta, "A Delaunay triangulation based node connection strategy for probabilistic roadmap planners," in *Proc. IEEE ICRA*, vol. 1. Apr./May 2004, pp. 908–913.

[163] A. M. Ladd and L. E. Kavraki, "Using motion planning for knot untangling," *Int. J. Robot. Res.*, vol. 23, nos. 7–8, pp. 797–808, Aug. 2004.

[164] E. Plaku, K. E. Bekris, B. Y. Chen, A. M. Ladd, and E. E. Kavraki, "Sampling-based roadmap of trees for parallel motion planning," *IEEE Trans. Robot.*, vol. 21, no. 4, pp. 597–608, Aug. 2005.

[165] K. Hauser, T. Bretl, J. C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *Int. J. Robot. Res.*, vol. 27, nos. 11–12, pp. 1325–1349, Nov./Dec. 2008.

[166] V. Vonasek, M. Saska, K. Kosnar, and L. Preucil, "Global motion planning for modular robots with local motion primitives," in *Proc. IEEE ICRA*, May 2013, pp. 2465–2470.

[167] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robot. Res.*, vol. 30, no. 12, pp. 1435–1460, Oct. 2011.

[168] X. Tang, S. Thomas, P. Coleman, and N. M. Amato, "Reachable distance space: Efficient sampling-based planning for spatially constrained systems," *Int. J. Robot. Res.*, vol. 29, no. 7, pp. 916–934, Jun. 2010.

[169] S. Dalibard, A. El Khoury, F. Lamiraux, A. Nakhaei, M. Taïx, and J.-P. Laumond, "Dynamic walking and whole-body motion planning for humanoid robots: An integrated approach," *Int. J. Robot. Res.*, vol. 32, pp. 1089–1103, Aug. 2013.

[170] E. Frazzoli, M. A. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1077–1091, Dec. 2005.

[171] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Trans. Robot.*, vol. 29, no. 1, pp. 105–117, Feb. 2013.
[172] A. Yershova and S. M. LaValle, "Improving motion-planning algorithms by efficient nearest-neighbor searching," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 151–157, Feb. 2007.
[173] N. M. Amato and L. K. Dale, "Probabilistic roadmap methods are embarrassingly parallel," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1. May 1999, pp. 688–694.
[174] J. Bialkowski, S. Karaman, and E. Frazzoli, "Massively parallelizing the RRT and the RRT*," in *Proc. IEEE/RSJ Int. Conf. IROS*, Jul. 2011, pp. 3513–3518.
[175] J. Canny, J. Reif, B. Donald, and P. Xavier, "On the complexity of kinodynamic planning," in *Proc. 29th Annu. Symp. Found. Comput. Sci.*, Oct. 1988, pp. 306–316.
[176] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic motion planning," *J. ACM*, vol. 40, no. 5, pp. 1048–1066, Nov. 1993.
[177] S. Sekhavat, P. Svestka, J.-P. Laumond, and M. H. Overmars, "Multilevel path planning for nonholonomic robots using semiholonomic subsystems," *Int. J. Robot. Res.*, vol. 17, no. 8, pp. 840–857, Aug. 1998.
[178] F. Lamiraux, D. Bonnafous, and O. Lefebvre, "Reactive path deformation for nonholonomic mobile robots," *IEEE Trans. Robot.*, vol. 20, no. 6, pp. 967–977, Dec. 2004.
[179] F. Lamiraux, E. Ferre, and E. Vallee, "Kinodynamic motion planning: Connecting exploration trees using trajectory optimization Methods," in *Proc. IEEE ICRA*, vol. 4. Apr./May 2004, pp. 3987–3992.
[180] F. Boyer and F. Lamiraux, "Trajectory deformation applied to kinodynamic motion planning for a realistic car model," in *Proc. IEEE ICRA*, May 2006, pp. 487–492.
[181] F. Lamiraux, J. P. Laumond, C. Van Geem, D. Boutonnet, and G. Raust, "Trailer truck trajectory optimization: The transportation of components for the Airbus A380," *IEEE Robot. Autom. Mag.*, vol. 12, no. 1, pp. 14–21, Mar. 2005.
[182] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *Int. J. Robot. Res.*, vol. 21, no. 3, pp. 233–255, Mar. 2002.
[183] T. Fraichard and H. Asama, "Inevitable collision states—A step towards safer robots?" *Adv. Robot.*, vol. 18, no. 10, pp. 1001–1024, Jan. 2004.
[184] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Motion planning with dynamics by a synergistic combination of layers of planning," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 469–482, Jun. 2010.
[185] E. Glassman and R. Tedrake, "A quadratic regulator-based heuristic for rapidly exploring state space," in *Proc. IEEE ICRA*, Feb. 2010, pp. 5021–5028.
[186] A. Shkolnik, M. Walter, and R. Tedrake, "Reachability-guided sampling for planning under differential constraints," in *Proc. IEEE ICRA*, May 2009, pp. 2859–2865.
[187] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," in *Proc. Amer. Control Conf.*, vol. 1. Jun. 2001, pp. 43–49.
[188] J. R. Bruce and M. M. Veloso, "Safe multirobot navigation within dynamics constraints," *Proc. IEEE*, vol. 94, no. 7, pp. 1398–1411, Jul. 2006.
[189] A. Shkolnik and R. Tedrake, "Path planning in 1000+ dimensions using a task-space Voronoi bias," in *Proc. IEEE ICRA*, May 2009, pp. 2061–2067.
[190] M. Kazemi, K. K. Gupta, and M. Mehrandezh, "Randomized kinodynamic planning for robust visual servoing," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1197–1211, Oct. 2013.
[191] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, "Bounding on rough terrain with the LittleDog robot," *Int. J. Robot. Res.*, vol. 30, no. 2, pp. 192–215, Feb. 2011.
[192] B. Lau, C. Sprunk, and W. Burgard, "Kinodynamic motion planning for mobile robots using splines," in *Proc. IEEE/RSJ Int. Conf. IROS*, Oct. 2009, pp. 2427–2433.
[193] E. Koyuncu and G. Inalhan, "A probabilistic B-spline motion planning algorithm for unmanned helicopters flying in dense 3D environments," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2008, pp. 815–821.
[194] K. Macek, M. Becked, and R. Siegwart, "Motion planning for car-like vehicles in dynamic urban scenarios," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 4375–4380.
[195] S. Karaman and E. Frazzoli, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *Proc. 49th IEEE CDC*, Dec. 2010, pp. 7681–7687.
[196] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, Jun. 2011.
[197] C. Guarino Lo Bianco, "Minimum-jerk velocity planning for mobile robot applications," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1317–1326, Oct. 2013.
[198] B. Akgun and M. Stilman, "Sampling heuristics for optimal motion planning in high dimensions," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2011, pp. 2640–2645.
[199] J. Nasir, F. Islam, U. Malik, Y. Ayaz, O. Hasan, M. Khan, *et al.*, "RRT*-SMART: A rapid convergence implementation of RRT*," *Int. J. Adv. Robot. Syst.*, vol. 10, pp. 1651–1656, Jun. 2013.
[200] A. H. Qureshi, K. F. Iqbal, S. M. Qamar, F. Islam, Y. Ayaz, and N. Muhammad, "Potential guided directional-RRT* for accelerated motion planning in cluttered environments," in *Proc. IEEE ICMA*, Aug. 2013, pp. 519–524.
[201] S. Choudhury, S. Scherer, and S. Singh, "RRT*-AR: Sampling-based alternate routes planning with applications to autonomous emergency landing of a helicopter," in *Proc. IEEE ICRA*, May 2013, pp. 3947–3952.
[202] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT*," in *Proc. IEEE ICRA*, May 2011, pp. 1478–1483.
[203] M. Elbanhawi and M. Simic, "On the performance of sampling-based optimal motion planners," in *Proc. 7th UKSim/AMSS Eur. Symp. Comput. Model. Simul.*, 2013, pp. 1–6.
[204] O. Arslan and P. Tsiotras, "Use of relaxation methods in sampling-based algorithms for optimal motion planning," in *Proc. IEEE ICRA*, May 2013, pp. 2421–2428.
[205] I. Ko, B. Kim, and F. C. Park, "VF-RRT: Introducing optimization into randomized motion planning," in *Proc. 9th ASCC*, Jun. 2013, pp. 1–5.
[206] G. Goretkin, A. Perez, R. Platt, and G. Konidaris, "Optimal sampling-based planning for linear-quadratic kinodynamic systems," in *Proc. IEEE ICRA*, Oct. 2013, pp. 2429–2436.
[207] D. J. Webb and J. van den Berg, "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics," in *Proc. IEEE ICRA*, May 2013, pp. 5054–5061.
[208] J. Jeong hwan, S. Karaman, and E. Frazzoli, "Anytime computation of time-optimal off-road vehicle maneuvers using the RRT*," in *Proc. 50th IEEE CDC-ECC*, Jun. 2011, pp. 3276–3282.
[209] S. Karaman and E. Frazzoli, "Sampling-based optimal motion planning for non-holonomic dynamical systems," in *Proc. IEEE ICRA*, Jan. 2013, pp. 5041–5047.
[210] P. Leven and S. Hutchinson, "A framework for real-time path planning in changing environments," *Int. J. Robot. Res.*, vol. 21, pp. 999–1030, Dec. 2002.
[211] L. Jaillet and T. Simeon, "A PRM-based motion planner for dynamically changing environments," in *Proc. IEEE/RSJ Int. Conf. IROS*, vol. 2. Sep./Oct. 2004, pp. 1606–1611.
[212] J. P. van den Berg, D. Nieuwenhuisen, L. Jaillet, and M. H. Overmars, "Creating robust roadmaps for motion planning in changing environments," in *Proc. IEEE/RSJ Int. Conf. IROS*, Aug. 2005, pp. 1053–1059.
[213] L. Tsai-Yen and S. Yang-Chuan, "An incremental learning approach to motion planning with roadmap management," in *Proc. IEEE ICRA*, vol. 4. Apr. 2002, pp. 3411–3416.
[214] R. Gayle, K. R. Klingler, and P. G. Xavier, "Lazy reconfiguration forest (LRF)—An approach for motion planning with multiple tasks in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 1316–1323.
[215] D. Ferguson, N. Kalra, and A. Stentz, "Replanning with RRTs," in *Proc. IEEE ICRA*, May 2006, pp. 1243–1248.
[216] J. van den Berg, D. Ferguson, and J. Kuffner, "Anytime path planning and replanning in dynamic environments," in *Proc. IEEE ICRA*, May 2006, pp. 2366–2371.
[217] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite RRTs for rapid replanning in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 1603–1609.
[218] N. A. Melchior and R. Simmons, "Particle RRT for path planning with uncertainty," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 1617–1624.
[219] H. Yifeng and K. Gupta, "RRT-SLAM for motion planning with motion and map uncertainty for robot exploration," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2008, pp. 1077–1082.
[220] A. Bry and N. Roy, "Rapidly-exploring Random Belief Trees for motion planning under uncertainty," in *Proc. IEEE ICRA*, Sep. 2011, pp. 723–730.

[221] M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "Path planning for motion dependent state estimation on micro aerial vehicles," in *Proc. IEEE ICRA*, May 2013, pp. 3926–3932.

[222] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and Gaussian processes," in *Proc. IEEE/RSJ Int. Conf. IROS*, Sep. 2008, pp. 1056–1062.

[223] S. Chakravorty and S. Kumar, "Generalized sampling-based motion planners," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 41, no. 3, pp. 855–866, Jun. 2011.

[224] A.-A. Agha-mohammadi, S. Chakravorty, and N. M. Amato, "FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements," *Int. J. Robot. Res.*, Nov. 2013.

MOHAMED ELBANHAWI received the B.Eng. degree in advanced manufacturing and mechatronics from RMIT University, Australia, in 2012, where he is currently pursuing the Ph.D. degree. His research interests include motion planning, car-like robots, and autonomous systems.

MILAN SIMIC received the B.E.E., M.E.E., and Ph.D. degrees in electronics from the University of Nis, Serbia, and Grad Dip.Ed. degree from RMIT University. He is a conducting management, research and teaching with RMIT University. Currently, he is a Lecturer of mechatronics, electronics and management with the School of Aerospace, Mechanical and Manufacturing Engineering. His main research interests are in digital communications, robotics, autonomous systems, and green energy.