Hindawi
Journal of Advanced Transportation
Volume 2017, Article ID 7816263, 10 pages
https://doi.org/10.1155/2017/7816263

Research Article

Liveness-Based RRT Algorithm for Autonomous Underwater Vehicles Motion Planning

Yang Li,[1] Fubin Zhang,[1] Demin Xu,[1] and Jiguo Dai[2]

[1] *The School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China*
[2] *Department of Mechanical Engineering, Texas Tech University, Lubbock, TX 79409, USA*

Correspondence should be addressed to Fubin Zhang; zhangfb@nwpu.edu.cn

Received 26 June 2017; Accepted 13 September 2017; Published 19 October 2017

Academic Editor: Cheng S. Chin

Motion planning is a crucial, basic issue in robotics, which aims at driving vehicles or robots towards to a given destination with various constraints, such as obstacles and limited resource. This paper presents a new version of rapidly exploring random trees (RRT), that is, liveness-based RRT (Li-RRT), to address autonomous underwater vehicles (AUVs) motion problem. Different from typical RRT, we define an index of each node in the random searching tree, called "liveness" in this paper, to describe the potential effectiveness during the expanding process. We show that Li-RRT is provably probabilistic completeness as original RRT. In addition, the expected time of returning a valid path with Li-RRT is obviously reduced. To verify the efficiency of our algorithm, numerical experiments are carried out in this paper.

1. Introduction

In the last decade, motion planning for single robot or multiple robot system has gained a lot of research interest as one of the primary problems in robot field. In most application cases, it mainly involves the structured mobility to drive the robots to the final destination given any initial states [1]. In general, the motion planning algorithms can be generally categorized into two types [2]. The first ones return the positions along the path from start to the goal, which are called path planning algorithms. In addition, the second ones, that is, trajectory planning algorithms (also called motion planning algorithms), return a sequence set of input space that drives the robot towards the destination.

In recent years, various algorithms have been proposed for both path and trajectory planning problems. Artificial potential fields (APF) can be viewed as a powerful method for path planning, which was firstly proposed in [3]. Although APF suffers from the presence of local minimums, it is adaptable in such cases with online intractable changes of environment. Meantime, various of methods are proposed to avoid sticking into local minimums, such as potential functions without local minimums [4] and random search methods [5–7].

Another remarkable probabilistic method was proposed as Probabilistic Roadmap (PRM) [8–11]. The key idea of PRM is to generate a connective random graph that comprises valid nodes selected randomly in the free configuration space (C-space). Meanwhile, RRT was presented as a single query incremental algorithm [12]. PRM and RRT are shown with more effectiveness, especially facing high-dimensional problems. Besides, they are insensitive to the complexity of the problems. RRT has provably probabilistic completeness, which means that the probability that RRT cannot find a valid path from the initial to the goal position converges to zero, as the number of samples tends to infinity. Also, the concept of probabilistic completeness is officially defined, which is a crucial property, although it is weaker than "completeness" in the field of sample based algorithms. Consequently, RRT has been applied in many applications ranging from industrial production to military battle systems.

Several RRT based algorithms are designed in order to deal with different constraints or to provide solutions more effectively. For example, in [13–16], additional dynamic differential constraints are considered when extending the random tree. A kind of new variation space, called trajectory parameter-space (TP-space), was presented in [17].

Particularly each edge is kinematically feasible once it is generated by two points in TP-space. In [18], the expanding process utilizes implicit flood-fill-like mechanism to avoid the local minimum. Thus, the expected time to find the solution is reduced. Because of the insensitivity to the dimension of problems, it is able to be used to search time invariant parameters [19]. It is a fact that AUVs have been widely applied in the marine resources exploration and utilization [20–23]. In order to map a scalar field, mutual information based multidimensional RRT algorithm was presented for multiple AUVs in [15].

However, RRT is proven to converge almost to a nonoptimal path [24], in which RRT* was detailed to be asymptotically optimal. That means RRT* will "almost surely" find the optimal path as the number of samples tends to infinity. RRT* applies the reconstruction of the searching tree, which always contains the optimal subtree at each iteration. Since then several RRT* based optimal planning algorithms were proposed [25–28] and applied in many fields [29, 30]. LBT-RRT was proposed in [25], which is a combination algorithm with RRT and RRT*. It defines a variable ϵ to describe the "distance" away from the optimal path. In other words, ϵ can be viewed as the comparison between RRT and RRT*. If ϵ equals zero, it becomes RRT* essentially. In [26], RRT* is refined to be capable of replanning with dynamic environments. In [27], the limitation, that is, requirements for an asymptotical optimal steer function during the tree's expansion, is provably addressed based on sparse data structure.

From [31], we know that the sampling strategies have much effect on the efficiency of sampling-based algorithms. In early researches, typical choice is sampling uniformly in C-space [8, 12, 24]. Every node in C-space is sampled with exactly the same probability, by which the prior information actually is not exploited. For instance, a new node intuitively prefers to be sampled away from the region where a large number of samples already exist. And it certainly reduces the probability that samples are generated in the case of narrow passages [31]. For this reason, various heuristic methodologies are designed. Goal bias is generally utilized to enhance extending towards the goal due to its effectiveness. Medial axis is combined with path planning algorithms to search the C-space with much more efficiency [8]. On the contrary, Gaussian sampling strategy prefers the region nearby obstacles [10, 32]. In this view, geometric shapes of obstacles are able to be quickly constructed. Some other hybrid sampling strategies are proposed to combine two or more methods. These methods uniformly consider the information of the environment to refine the sampling probability; however, the information of nodes is not totally utilized. Specifically, regions where a number of nodes are already sampled can be mostly considered with less worth. In other words, it is preferred to sample in "new" regions.

In this paper, we demonstrate the expanding mechanism of the random tree and make an estimation of the expanding ability of each node. We present several new indicators in the expanding procedure, which are utilized in our new algorithm. Also, we detail the indicators quantitative descriptions of the nodes. Meanwhile, we provide theoretical analysis and property of our algorithm with mathematical proof (i.e., probabilistic completeness and outperforming compared to RRT). The proof is based on a technical notation attraction sequence and probability theory.

The reminder of the paper is organized as follows. Section 2 introduces the definition and notations in this paper. The indicators which reflect the growing of the tree and expanding ability are analyzed in Section 3. Then our new algorithm, that is, Li-RRT, is described in detail. In Section 4, the analysis of Li-RRT is provided. Section 5 presents an application example for a planar mobile robot. In final, concluding remarks are drawn in Section 6.

2. Preliminaries and Problem Formation

2.1. Problem Formulation. Path planning problem can be generally viewed as a search problem in the C-space $C \subseteq \mathbb{R}^n$. For instance, the C-space of a planar two-wheel mobile robot can be defined as $C \subseteq \mathbb{R}^3$, that is, position (x, y), heading angle θ. In this way, any $x \in C \subseteq \mathbb{R}^3$ can describe the state of the mobile robot. Additionally, we define $C_{\text{obs}} \subset C$ and $C_{\text{free}} = C \setminus C_{\text{obs}}$ as the obstacle C-space and free C-space separately. The path planning problems require a finite sequence of states in C_{free} from an initial state to a given goal state. Here we present the basic path planning problem as defined in [24].

Problem 1 (path planning problem [24]). The bounded C-space C, obstacle space C_{obs}, initial state $x_{\text{start}} \in C_{\text{free}}$, goal region $X_{\text{goal}} \in C_{\text{free}}$, denoted as tuple $\{x_{\text{start}}, C_{\text{free}}, X_{\text{goal}}\}$, return a path $\xi : [0, 1] \to C_{\text{free}}$, such that $\xi(0) = x_{\text{start}}$, $\xi(1) \in X_{\text{goal}}$. If there exist no feasible paths, return failure.

Facing path planning problems, each element of dimension is considered independently with no other constraints. However, it is unrealistic in most applications. Specifically, the kinodynamic of AUVs is restricted in case of uncontrollability. It means that there exists corresponding available input sequence $u(s) : [0, 1] \to U$ that drives the AUV from x_{start} to X_{goal}. Subsequently the path can also be viewed as a function of input sequence $\xi(u(\cdot)) : u(\cdot) \to C_{\text{free}}$. Formally, we give the definition of trajectory planning problem for AUVs.

Problem 2 (trajectory planning problem for AUVs). Given $\{x_{\text{start}}, C_{\text{free}}, X_{\text{goal}}\}$ and kinodynamic constraint of AUVs $\dot{x} = f(x, u)$, return an available control sequence $u : [0, 1] \to U$, such that the path $\xi(u(\cdot)) = \{x \in C_{\text{free}} \mid \xi(u(0)) = x_{\text{init}}, \xi(u(1)) \in C_{\text{free}}, \xi(u(s)) = \int_0^1 [x_{\text{init}} + f(x, u)]ds\}$.

In this paper, we focus on the trajectory planning problem for AUVs. Since the initial force has major impact on the motion of AUVs compared to ground vehicles, the kinodynamic constraints are more complex.

3. Algorithms

RRT was firstly introduced by Lavalle as a sampling-based path planning algorithm, which is shown in Algorithm 1. It has the ability of quickly searching or exploring the C-space by a random tree. At each iteration, the tree expands

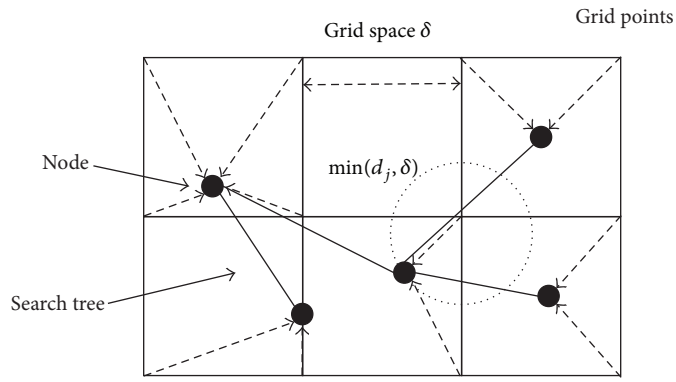


Figure 1: Grid point distance from the nearest node in the tree. δ is an adjustable grid spacing, and the dashed arrow indicates the distance, d_j, from the grid points to their closet nodes in tree.

Require: C-space C, Free C-space C_{free}, Initial state x_{start}, Goal region X_{goal}, max searching iteration K;
Ensure: Feasible path τ
(1) $V \leftarrow (x_{\text{start}})$; $E \leftarrow \emptyset$; $\mathcal{G} \leftarrow (V, E)$; // Initialization
(2) for $k = 1, 2, \ldots, K$ do
(3) $x_{\text{rand}} \leftarrow$ Sample(k); // Randomized sample in C_{free}
(4) $\mathcal{G}(V, E) \leftarrow$ Extend($\mathcal{G}$, x_{rand}); // Extend random tree
(5) end for
(6) $\tau \leftarrow$ Export($\mathcal{G}$); // Return the feasible path
(7) return τ;

Algorithm 1: Body of basic RRT algorithm [33].

sequentially towards the goal. We refer readers to more details about RRT in [12].

The main branches of RRT are firstly constructed as it rapidly reaches the far corners of the square. As samples are incrementally added in the searching tree, C-spaces are mostly covered by smaller branches gradually. In other words, given any interval $\epsilon > 0$, C-space will be filled with nodes as the number of samples tends to infinity. It means the probability that there exists at least one node of the random tree located in the goal set equals one, as the number of samples tends to infinity, if there exists one available path in C_{free}: that is, $\lim_{n\to\infty} \mathbb{P}\{\forall x \in \mathcal{C}, \exists x' \in x_{\text{RRT}}, |x - x'| < \epsilon, \epsilon > 0\} = 1$.

3.1. Coverage of a Graph. A kind of coverage measure of the tree in C-space was proposed in [34]. We uniformly overlay grids of n_g points and spacing δ on C. Define the minimum distance from each grid point j to the nearest node in the tree as d_j. Thus, $\min(d_j, \delta)$ can describe the radius of the largest ball centered at the grid point which contains no nodes of the tree and adjacent grid points as shown in Figure 1. Given a graph $\mathcal{G}$, we define its coverage measure as in [34].

$$c(\mathcal{G}) = \frac{1}{\delta} \sum_{j=1}^{n_g} \frac{\min\left(d_j, \delta\right)}{n_g}, \tag{1}$$

where $1/\delta$ removes the impact of different spacing distance. The coverage of a tree can be viewed as the average of all nodes distances, normalized by the grid space δ.

The key idea of this measure is to describe the dispersion of all nodes, which is a conception proposed from the Monte Carlo method that indicates the unevenness of sample points. Primarily, the cover performance is significantly better with smaller coverage measure. In other words, expanding more widely and uniformly may lead to smaller coverage measure. In an extreme case, that is, $c(\mathcal{G}) = 0$ which is impracticable, if there exists at least one node of the tree in every grid point, the tree can be viewed well distributed in C_{free} with given solution of the grids. Intuitively, we may try to optimize $c(\mathcal{G})$ with an acceptable computing complexity, which is addressed in Section 3.4.

3.2. Growth Speed of a Tree. The growth of a tree is naturally defined as the derivative of the coverage measure. We denote the number of nodes in the tree as n_t and then define the growth of the tree as [34]

$$g(\mathcal{G}) = \frac{-\Delta c(\mathcal{G})}{\Delta n_t}. \tag{2}$$

Because the differential form $dc(\mathcal{G})/dn_t$ is intractably formed and computed, we utilize the difference form to indicate the growing speed of the tree. Obviously, $g(\mathcal{G})$ is able to be used to test the nodes of tree appropriately. Let $\overline{g}$ denote the threshold, if $g(\mathcal{G})$ drops below $\overline{g}$ with new node v_i, it means that v_i is almost worthless.

3.3. Liveliness of Nodes. The factors which illustrate expanding ability of nodes in a tree can be roughly categorized into two parts: the external circumstance factors and the internal circumstance factors.

3.3.1. External Circumstance of Nodes. External environment has significant influence on the expanding ability of each node. If some node locates in a blind alley, it has absolutely no contribution on searching. As shown in Figure 2, it describes the scope of a parent node that could be extended to within

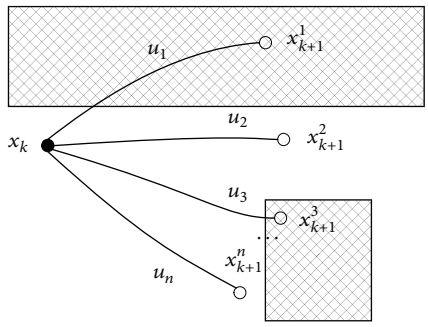


Figure 2: Latent expansion with obstacles.

time interval Δt. Without loss of generality, we assume that an appropriate discrete control set U is defined and the cardinality of U is m: that is, $U = u_1, u_2, \ldots, u_m$. Let x_k denote the parent and let $x_{k+1}^{u_i}$, $i = 1, \ldots, m$ denote the successor of x_k. The superscript u_i means the control vector applied on x_k, where the gray areas represent the obstacles. If $x_{k+1}^{u_i}$ locates in C_{obs} for all $k = 1, 2, \ldots, m$, it certainly means that x_k is dead for the concurrent tree. Also, this expansion by x_k should be considered totally useless. By reducing useless nodes the searching process can be obtained more efficiently.

Based on that, the available potential successor set of x_k is achieved by applying U; that is, $\widetilde{X}_{k+1}^U = \{x_{k+1}^{u_i} \mid \text{Collision\_Free}(x_k, x_{k+1}^{u_i}) = \text{TRUE}, i = 1, 2, \ldots, m\}$. Meantime, we denote the corresponding available control vector as $\widetilde{U} = \{u_i \mid \text{Collision\_Free}(x_k, x_{k+1}^{u_i}) = \text{TRUE}, i = 1, 2, \ldots, m\}$. We denote the cardinality of $\widetilde{U}$ as m_a and define collision detective index as $\text{Se}(x_k) = m_a/m$. Note that if $\text{Se}(x_k) = 0$, the point x_k is a dead point. If so, x_k is necessarily removed from the tree. Meanwhile, we could define the collision detective index of a tree as

$$\text{Se}(\mathcal{G}) = \frac{\sum_{k=1}^{n_l} \text{Se}(x_k)}{n_l}, \tag{3}$$

where n_l denotes the number of nodes in a tree, which is utilized to normalize the feature of the nodes. The collision detective index of a tree is the average value of all component nodes.

It is a fact that the prior information about the environment is unavailable in many applications. In such cases, the performance of searching C-space is more important. $\text{Se}(\mathcal{G})$, defined in (3), may give a reasonable solution on the process of expansion of searching tree. Generally, we prefer the nodes that make $\text{Se}(\mathcal{G})$ as large as possible or larger than a threshold $\overline{Se}$, because it means that the probability of reaching the target will stay high.

Unfortunately, it is commonly difficult to tell whether a node is stuck in a blind alley or just in a narrow passage. Note that it is still necessary to expand x_k if m_a is not equal to zero. In other words, only $\{x_k \mid m_a = 0\}$ could be removed in case of unexpected failure. In this way, the set of state $\{x_k \mid m_a = 0\}$ can be viewed as the inevitable obstacles. Unlike that, the dead nodes in this paper are naturally cheap to compute.

3.3.2. Internal Structure of Trees. Internal structure of a tree also influences the expanding ability of nodes. Due to the stochastic sampling strategy, it may lead to crowd in some regions by random expanding. In Figure 3(a), the gray circles with dotted lines are considered as regression nodes, which are avoided by RRT-blossom method proposed in [18]. Although it may be necessary to explore the regression states since the complex environment, the other unexplored regions have reasonable high priority.

In Figure 3(a), the hollow white dots are possible newly added nodes of an identical parent. The hollow white dots embraced in the ellipse together with a solid black dot are closer to the neighbor nodes than their parents. Figure 3(b) shows all expansions in the tree which avoid being closer to the existing structure. This strategy ensures that the tree will have a strong tendency to explore the whole C-space with inflated branches.

We define the degree of a node as the number of its successors, which is denoted by $d(x_k)$. All offspring of a node is defined as

$$ds(x_k) = \sum_{x_i \in \text{Us}(x_k)} d(x_i), \tag{4}$$

where the set $\text{Us}(x_k)$ is the union of all offspring of the node x_k. It is obviously that a node with more successors and offspring has less expanding ability. Figure 4 illustrates a tree's distribution of degrees and offspring.

The neighbors of a node are viewed as obstacles, which are shown in Figure 5. The neighbor nodes are viewed as circular or spherical obstacles with a fixed radius. Define the number of neighbor collisions as $m_a^{\text{nei}}(x_k) = \sum_{i=1}^{m} 1\{\text{Collision\_Free}(x_k, x_{k+1}^{u_i} = \text{TRUE})\}$, where $1(\cdot)$ equals 1 if equation $(\cdot)$ holds.

Considering all these factors, we design a hybrid indicator to illustrate the quantity of liveliness of each node. Synthesizing two external and internal factors, the liveliness index of each node in a tree is defined in the form of

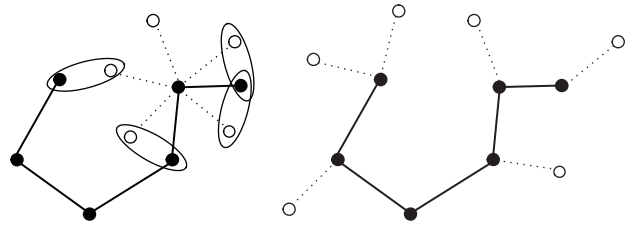
$$\begin{aligned}&\text{Li}(x_k)\\ &= \text{Se}(x_k) \exp(-m_h) \exp[-d(x_k)] \exp\left[-\frac{ds(x_k)}{n_t}\right],\end{aligned} \tag{5}$$

where n_t is the number of total nodes at current iteration. Indeed, (5) reflects the expanding ability of a specific node. We translate $\text{Li}(x_k)$ in logs denoted as LnLi.

$$\begin{aligned}\text{LnLi}(x_k) &= \ln \text{Li}(x_k)\\ &= \ln \text{Se}(x_k) - \left[m_h + d(x_k) + \frac{ds_k}{n_t}\right].\end{aligned} \tag{6}$$

Since log function has the same monotonicity, $\text{LnLi}(x_k)$ can describe the liveness of each node. Particularly, if x_k is a dead point, that is, $\text{Se}(x_k) = 0$, we set $\text{LnLi}(x_k) = -\inf$.

Obviously, the liveness LnLi defined in (6) will be attached to each node once added to the random tree. The value of LnLi is calculated and updated when the expanding process will be called. It can be viewed as a guidance that describes a better expanding direction for the searching tree.



(a) Regression expansions (b) Nonregressing expansions

Figure 3: RRT-blossom regression. New point tends to stay away from the existing structure, like the hollow circles.

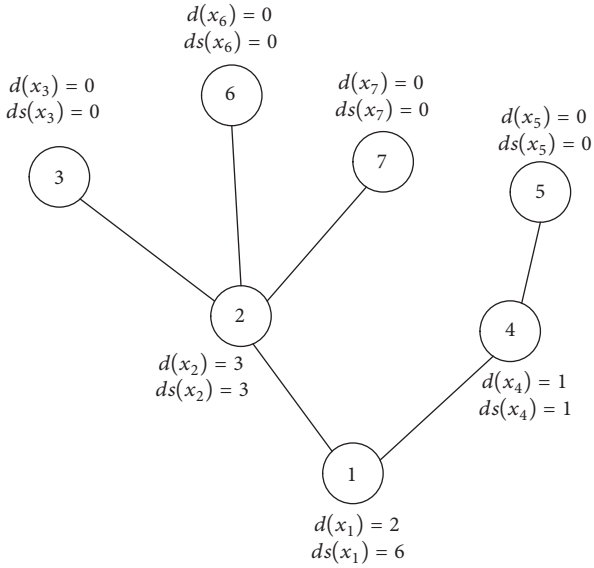


Figure 4: The degree of nodes in a tree.

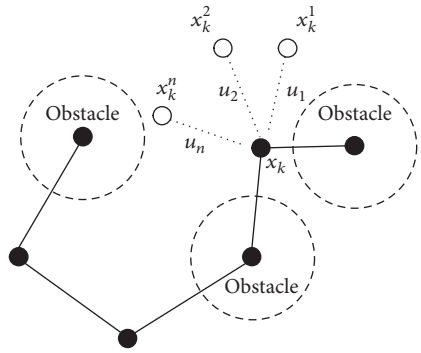


Figure 5: Neighbor obstacles. The radius of neighbor obstacle is generally obtained from experience.

3.4. Li-RRT. In this section, we present the hybrid RRT algorithm, that is, Li-RRT, as shown in Algorithm 2. It utilizes the liveness of each node to guide the expanding process of the random searching tree. More efficient or useful nodes will be popped out to enhance the property of exploration. The main methodology of Li-RRT is based on RRT. Different from the typical RRT, we define a liveness set $L = \{\mathrm{LnLi}(v) \mid v \in V\}$ initially (line (1)).

At each iteration, node max Li pops with max liveness from L in line (3). Then an input vector is randomly selected from function RandU (line (5)), according to which a random state in C_{free} is generated. Meantime, x_{rand} is chosen from X_{goal} with probability $\mathscr{W}$ (line (13)). It is proposed as a goal bias sampling method, which is simple but effective. Similarly, the nearest node is calculated given the cost function $c(\cdot)$. New node x_{new} then is achieved based on forward processing drive (line (20)). CollisionFree is called to verify the validity of x_{new} and $(x_{\mathrm{nearest}}, x_{\mathrm{new}})$ (line (21)). Expanding process is implemented by adding x_{new} and $(x_{\mathrm{nearest}}, x_{\mathrm{new}})$ (line (22)).

Liveness of each node is updated by UpdateLi(v) in Algorithm 3. From the definition of $\mathrm{LnLi}(x_k)$ (see (5)), we only need to care about three nodes sets, that is, $\{V_{\mathrm{ch}}^1, V_{\mathrm{ch}}^2, V_{\mathrm{ch}}^3\}$. V_{ch}^1 contains only x_{new}, since $\mathrm{Se}(x_k)$ is constant and is determined based on the state and the environments. Here we define that "all parent nodes" of x_k are the nodes set $\{v \mid x_k \text{ can be traced from } v\}$ different from the parent node $x_{\mathrm{par}}(x_k)$. V_{ch}^2 includes exactly all parent nodes of x_{new}. Because x_{new} increases the successors of $v \in V_{\mathrm{ch}}^2$, we set $\mathrm{LnLi}(v \in V_{\mathrm{ch}}^2) = \mathrm{LnLi}(v) + (ds(v)/n_t - (ds(v) + 1)/(n_t + 1))$. Specifically, we set the liveness of $x_{\mathrm{par}x}(x_{\mathrm{new}})$ as $\mathrm{LnLi}(x_{\mathrm{par}x}) = \mathrm{LnLi}(x_{\mathrm{par}x}) - 1$. V_{ch}^3 is the near set of x_{new}: that is, $V_{\mathrm{ch}}^3 = \{v \mid c(v, x_{\mathrm{new}}) \le \mathscr{H}\}$. m_h of $v \in V_{\mathrm{ch}}^3$ will increase because of x_{new}. Thus, we define $\mathrm{LnLi}(v \in V_{\mathrm{ch}}^3) = \mathrm{LnLi}(v) - 1$.

4. Analysis of the Algorithms

To prove the following theorems, we remind the definition of attraction sequence [35]. Let $\mathbb{A} = \{A_0, A_1, \ldots, A_k\}$ be a finite sequence of sets as follows. (i) For each A_i, there exists a set $A_i \subseteq B_i \subseteq C_{\mathrm{free}}$ called the basin of A_i, and $\forall x \in A_{i-1}$ and $y \in A_i$ and $z \in C_{\mathrm{free}} \setminus B_i$, $\|x - y\| \le \|x - z\|$ holds. (ii) $\forall x \in B_i$, there exists an m such that the sequence of action $\{u_1, u_2, \ldots, u_m\}$ selected by LOCAL_PLANNER algorithm will bring the point into $A_i \subseteq B_i$. (iii) $A_0 = \{x_{\mathrm{start}}\}$ and $A_k = \{X_{\mathrm{goal}}\}$.

An attractor region A_i like a funnel as a metaphor for motions converges into a small region in the space. As shown in Figure 6, a basin B_i can be viewed as a safety zone which ensures that a point of B_i will be selected by the NEAREST_NEIGHBOR and potential well which attracts the point into A_i. Given an attraction sequence $\mathbb{A}$ with length k and letting $p_i = \mu(A_i)/\mu(C_{\mathrm{free}})$ and $p_m = \min p_i$, we have the following lemma.

Require: C-space C, Obstacle space C_{obs}, Initial point x_{start}, Goal region X_{goal}, Discrete input set $U = [u_1, u_2, \ldots, u_f]$, Maximum search steps K;
Ensure: Feasible trajectory ξ
(1) $V \leftarrow x_{start}$; $E \leftarrow \emptyset$; $\mathcal{G} \leftarrow (V, E)$; Liveness set $L = \{\mathrm{Li}(x_{start})\}$
// Initialization
(2) for $k = 1, \ldots, K$ do
(3) max Li $\leftarrow$ argmax$_{v \in V}$LnLi(v)
(4) if Rand() $< \mathscr{B}$ then
(5) $u_{rand} \leftarrow$ RandU($u_1, \ldots, u_f$)
(6) $x_{rand} \leftarrow$ drive(max Li, u_{rand}); $x_{nearest} \leftarrow$ max Li
(7) for each $x_{near} \in X_{near}$ do
(8) if $c(x_{near}, x_{rand}) < c(\max \mathrm{Li}, x_{rand})$ then
(9) $x_{nearest} \leftarrow x_{near}$
(10) end if
(11) end for
(12) else
(13) if Rand() $< \mathscr{W}$ then
(14) $x_{rand} \leftarrow x_{goal} \in X_{goal}$
(15) else
(16) $x_{rand} \leftarrow$ RandProc(C_{free})
(17) $x_{nearest} \leftarrow$ Nearest(x_{rand})
(18) end if
(19) end if
(20) $x_{new} \leftarrow$ drive($x_{nearest}, x_{rand}$)
(21) if CollisionFree($x_{nearest}, x_{rand}$) then
(22) $V \leftarrow V \cup x_{new}$; $E \leftarrow E \cup (x_{nearest}, x_{new})$
(23) end if
(24) if $x_{new} \in X_{goal}$ then
(25) objective = TRUE
(26) end if
(27) for all $v \in V$ do
(28) UpdateLi(v)
(29) end for
(30) end for
(31) return ξ if objective = TRUE; else failure

Algorithm 2: Liveliness-based RRT algorithm.

Require: Near set threshold $\mathscr{K}$; Three nodes set $\{V_{ch}^1, V_{ch}^2, V_{ch}^3\}$
(1) LnLi(x_{new}) $\leftarrow \ln \mathrm{Se}(x_{new}) + [-m_h(x_{new})]$
(2) for $v \in V_{ch}^2$ do
(3) LnLi(v) $\leftarrow$ LnLi(v) + $(ds(v)/n_t - (ds(v) + 1)/(n_t + 1))$
(4) end for
(5) $x_{par} \leftarrow$ parent(x_{new})
(6) LnLi(x_{par}) $\leftarrow$ LnLi(x_{par}) $- 1$
(7) for $v \in V_{ch}^3 =$ Near(x_{new}) do
(8) LnLi(v) $\leftarrow$ LnLi(v) $- 1$
(9) end for

Algorithm 3: Update liveness algorithm.

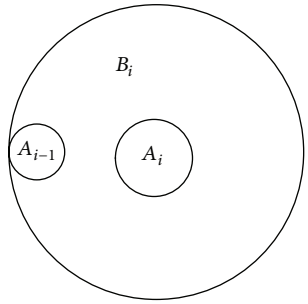


Figure 6: Attraction sequence.

Lemma 3 (see [36]). *Let $C_1, C_2, \ldots, C_n$ be independent Poisson trials such that, for $1 \le i \le n$, $\Pr[C_i = 1] = p_i$, where $0 < p_i < 1$. Then, for $\mathscr{C} = \sum_{i=1}^{n} C_i$, $\mu = \mathrm{E}[\mathscr{C}] = \sum_{i=1}^{n} p_i$, and $0 < \delta \le 1$,*

$$\Pr\left[\mathscr{C} < (1-\delta)\mu\right] < \exp\left(-\frac{\mu\delta^2}{2}\right). \tag{7}$$

Lemma 4 (see [35]). *If an attraction sequence of length k exists, for a constant δ, the probability of basic RRT algorithm fails to find a trajectory after n iterations are at most* $\exp[-(1/2)(np_m - 2k)]$.

Let sequence $\{p_1', p_2', \ldots, p_n'\}$ be an ascending order that $p_1' \le p_2' \le \cdots \le p_n'$, where $p_i' \in \{p_1, p_2, \ldots, p_n\}$, $p_1' = \min_i p_i$ and $p_n' = \max_i p_i$; the following theorem can be achieved.

While C_{free} is partitioned into m connected regions, motion planning problem for multiple AUVs could be considered as an m goals version of Problem 2. $\mathbb{A}^{(i)} = \{A_1^{(i)}, A_2^{(i)}, \ldots, A_{k_i}^{(i)}\}$ is an attraction sequence which connects x_{start} and $X_{i\mathrm{goal}}$, where k_i is the length of attraction sequence. Let $p_j^{(i)} = \mu(A_j^{(i)})/\mu(C_{\mathrm{free}})$, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, k_i$, $p_m = \min_{ij} p_j^{(i)}$; we have the following theorem.

Theorem 5. *If m attraction sequences of length k_i, $i = 1, 2, \ldots, m$ exist, the probability of basic RRT algorithm fails to find m trajectories after n iterations are at most* $\exp[-(1/2)(np_m - 2\sum_{i=1}^{m} k_i)]$.

Theorem 6. *If m attraction sequences of length k_i, $i = 1, 2, \ldots, m$ exist, the probability of liveliness-based RRT algorithm fails to find m trajectories after n iterations are at most* $\exp[(1/n)\sum_{i=1}^{m}\sum_{j=1}^{n_i}((p_j'^{(i)}/p_m)(-(1/2)np_m + \sum_{i=1}^{m} k_i))]$.

Proof. Let sequence $(p_1'^{(i)}, p_2'^{(i)}, \ldots, p_{n_i}'^{(i)})$ be m ascending orders satisfying $(p_1'^{(i)} \le p_2'^{(i)} \le \cdots \le p_{n_i}'^{(i)})$, where $p_1'^{(i)} = \min_j p_j'^{(i)}$, and $p_{n_i}'^{(i)} = \max_j p_j'^{(i)}$, $j = 1, 2, \ldots, n_i$, $i = 1, 2, \ldots, m$. While using the strategy of eliminating the "useless" nodes, the random variable $\mathscr{C}$ is viewed as Poisson trails with probability distribution $(p_j'^{i})$, $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n_i$. $\mathscr{C}$ is viewed as Poisson trails with probability distribution $p_j'^{(i)}$; then $\mu = \mathrm{E}[\mathscr{C}] = \sum_{i=1}^{m}\sum_{j=1}^{n_i} p_j'^{(i)}$, and $0 < \delta \le 1$. According to Lemma 3, $\Pr[\mathscr{C} < (1-\delta)\mu] < \exp(-\mu\delta^2/2)$, where $\delta = 1 - \sum_{i=1}^{m} k_i/(np_m)$. In addition,

$$\begin{aligned}
\frac{-\mu\delta^2}{2} &= -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n_i}\frac{p_j'^{i}}{p_m}\left(1 - \frac{\sum_{i=1}^{m} k_i}{np_m}\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i}\frac{p_j'^{i}}{p_m}\left(-\frac{1}{2}np_m + \sum_{i=1}^{m} k_i - \frac{\left(\sum_{i=1}^{m} k_i\right)^2}{2np_m}\right) \\
&< \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i}\frac{p_j'^{i}}{p_m}\left(-\frac{1}{2}np_m + \sum_{i=1}^{m} k_i\right).
\end{aligned} \tag{8}$$

This completes the proof. □

Proposition 7. *Given specific C, C_{free}, x_{start}, and a set of goal regions $\{X_{1\mathrm{goal}}, X_{2\mathrm{goal}}, \ldots, X_{m\mathrm{goal}}\}$, while solving Problem 2, the liveliness-based RRT algorithm always has smaller failing probability than the basic RRT algorithm.*

Proof. Note that $(1/n)\sum_{i=1}^{m}\sum_{j=1}^{n_i}(p_j'^{i}/p_m) > 1$, according to Theorems 5 and 6,

$$\begin{aligned}
\Pr[\mathrm{LiRRT}] &= \exp\left[\frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i}\frac{p_j'^{(i)}}{p_m}\left(-\frac{1}{2}np_m + \sum_{i=1}^{m} k_i\right)\right] \\
&< \exp\left[-\frac{1}{2}\left(np_m - 2\sum_{i=1}^{m} k_i\right)\right] \\
&= \Pr[\mathrm{bRRT}].
\end{aligned} \tag{9}$$

□

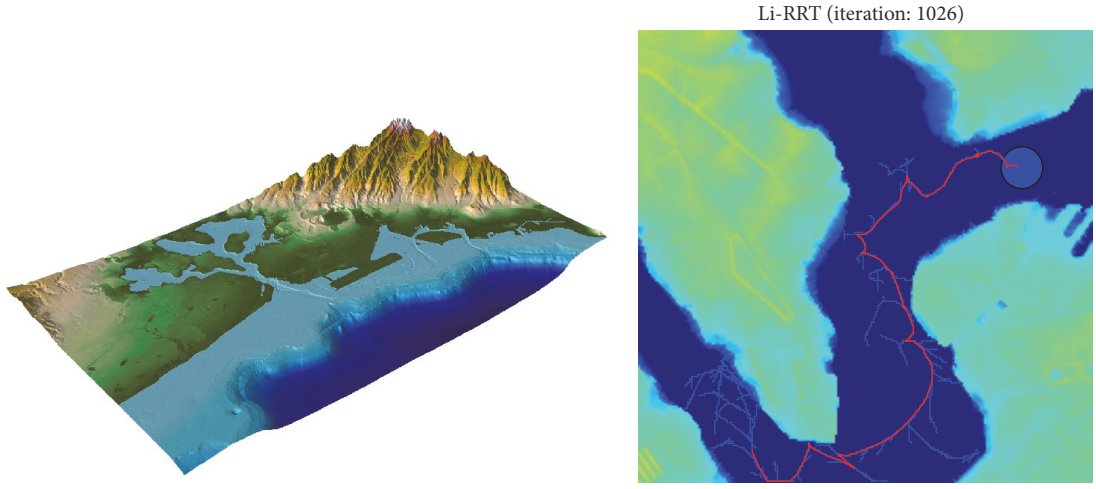
In summary, we can see that Li-RRT enhances the expanding efficiency from Proposition 7.

5. Applications of Li-RRT

In this section, we present a numerical simulation for planar AUV with data from NOAA. A region of Hawaii, that is, a rectangle from $157^\circ 58' 48'' \mathrm{W}$, $21^\circ 20' 24'' \mathrm{N}$ to $157^\circ 57' 36'' \mathrm{W}$, $21^\circ 21' 36'' \mathrm{N}$, is utilized to testify the effectiveness of Li-RRT; see Figure 7(a). Without loss of generality, we assume that AUV only voyages at depth of 5 meters at least. And the dynamics of AUV is $\dot{x} = v\cos(\theta)$, $\dot{y} = v\sin(\theta)$, $\dot{\theta} = \omega$. Meantime, we set $0 \le |v| \le 2$ m/s and $0 \le |\omega| \le 0.15$ rad/s.

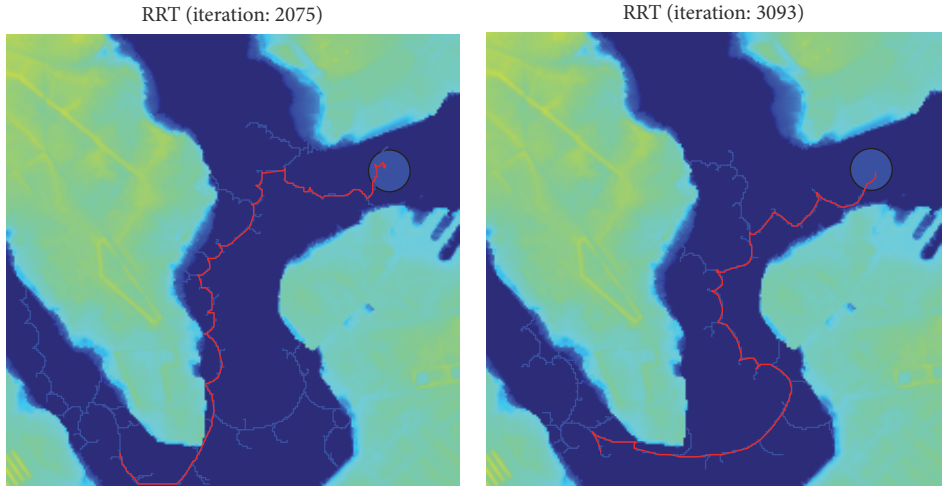
We can see that a valid path is found by Li-RRT shown as in Figure 7(b). Although solutions returned by Li-RRT are mostly not optimal, it can supply candidate paths effectively in less planning time. A comparison between RRT and Li-RRT is described in Figure 8. Li-RRT obviously costs less searching iterations by utilizing existing information between current random tree and the environment.

Here we take multiple simulations by RRT with same conditions and present two better solutions. We can see that RRT needs more searching iterations than Li-RRT averagely. Sampling strategy in typical RRT utilizes only goal bias method which tries to make the random searching



(a) Hawaii DEM figure (b) Path found by Li-RRT

FIGURE 7: Start position locates at 157°58′30″W, 21°20′29″N; goal locates at 157°57′47″W, 21°21′14″N, shown as a solid blue circle. Red line describes a valid path from start to goal.



(a) RRT with 2075 iterations (b) RRT with 3093 iterations

FIGURE 8: Planning with the same conditions by RRT. Start position locates at 157°58′30″W, 21°20′29″N; Goal locates at 157°57′47″W, 21°21′14″N, shown as a solid blue circle. Red line describes a valid path from start to goal.

tree towards the goal region. It certainly may generate useless sampling nodes in every searching step. For example, sampling that avoids an oversampled areas seems more efficient; however, typical RRT has no such ability. Thus, RRT needs more searching time or iterations to reach the goal.

6. Conclusions

In this paper, we have presented an analysis of rapidly exploring random trees and defined the several indicators of a tree, including the coverage and the growth speed describing the growing behavior of a tree. Considering the external and internal factors which influence the expanding ability, we also have defined indicators collision detective index, degree, offspring, and neighbor collisions. On this basis, we have proposed Li-RRT for AUVs motion planning. By using the tools attraction sequence, theoretical analysis has indicated that liveliness-based RRT enhances the expanding speed. Simulations are also provided to show the effectiveness of Li-RRT. It has been shown that Li-RRT performs well in finite planning time. Moreover, the growth direction of a tree also reflects some properties of a tree's expanding ability which could be utilized by RRT to conduct the expanding procedure. One of the possible research directions will focus on the planning without any prior information in the uncertain environment considering such indicators proposed in this work.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 61472325 and 61273333.

References

[1] Y. J. Heo and W. K. Chung, "RRT-based path planning with kinematic constraints of AUV in underwater structured environment," in *Proceedings of the 10th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI '13*, pp. 523–525, November 2013.

[2] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: a general overview," *Mechanisms and Machine Science*, vol. 29, pp. 3–27, 2015.

[3] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '85)*, vol. 2, pp. 500–505, 1985.

[4] S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 5, pp. 615–620, 2000.

[5] C. M. Saaj, V. Lappas, and V. Gazi, "Spacecraft swarm navigation and control using artificial potential field and sliding mode control," in *Proceedings of the 2006 IEEE International Conference on Industrial Technology, ICIT '06*, pp. 2646–2651, 2007.

[6] P. F. J. Lermusiaux, T. Lolla, P. J. H et al., *Science of Autonomy: Time-Optimal Path Planning and Adaptive Sampling for Swarms of Ocean Vehicles*, Springer International Publishing, Gewerbestrasse, Switzerland, 2016.

[7] S. Carpin and G. Pillonetto, "Merging the adaptive random walks planner with the randomized potential field planner," in *Proceedings of the in Proceedings of the Fifth International Workshop on Robot Motion and Control, 2005*, pp. 151–156, 2005.

[8] 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 *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1024–1031, May 1999.

[9] L. Han and N. M. Amato, "A kinematics-based probabilistic roadmap method for high DOF closed chain systems," *IEEE International Conference on Robotics and Automation*, vol. 1, pp. 473–478, 2004.

[10] V. Boor, M. H. Overmars, and A. F. V. D. Stappen, "Gaussian sampling strategy for probabilistic roadmap planners," in *Proceedings of the 1999 IEEE International Conference on Robotics and Automation, ICRA '99*, pp. 1018–1023, May 1999.

[11] R. Cui, B. Gao, and J. Guo, "Pareto-optimal coordination of multiple robots with safety guarantees," *Autonomous Robots*, vol. 32, no. 3, pp. 189–205, 2012.

[12] S. M. LaValle, "Rapidly-exploring random trees: a new tool for path planning," Tech. Rep. 98-11, Computer Science Dept., Iowa State University, Iowa, Iowa, USA, 1998.

[13] J. Kim and J. P. Ostrowski, "Motion planning a aerial robot using rapidly-exploring random trees with dynamic constraints," in *Proceedings of the in IEEE International Conference on Robotics and Automation, 2003*, vol. 2, pp. 2200–2205, 2003.

[14] D. Shim, H. Chung, H. J. Kim, and S. Sastry, "Autonomous exploration in unknown urban environments for unmanned aerial vehicles," in *Proceedings of the in AIAA GNC Conference*, pp. 1–8, 2005.

[15] R. Cui, Y. Li, and W. Yan, "Mutual information-based multi-auv path planning for scalar field sampling using multidimensional RRT," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 7, pp. 993–1004, 2016.

[16] T. Van Huynh, M. Dunbabin, and R. N. Smith, "Convergence-guaranteed time-varying RRT path planning for profiling floats in 4-dimensional flow," in *Proceedings of the Australasian Conference on Robotics and Automation, ACRA '14*, December 2014.

[17] J. L. Blanco, M. Bellone, and A. Gimenez-Fernandez, "TP-space RRT - Kinematic path planning of non-holonomic any-shape vehicles," *International Journal of Advanced Robotic Systems*, vol. 12, article no. A55, 2015.

[18] M. Kalisiak and M. van de Panne, "RRT-blossom: RRT with a local flood-fill behavior," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '06)*, pp. 1237–1242, May 2006.

[19] J. M. Esposito, J. Kim, and V. Kumar, *Adaptive RRTs for Validating Hybrid Robotic Control Systems*, Springer, Berlin, Germany, 2005.

[20] J. Si and C. Chin, "An adaptable walking-skid for seabed ROV under strong current disturbance," *Journal of Marine Science and Application*, vol. 13, no. 3, pp. 305–314, 2014.

[21] R. Cui, L. Chen, C. Yang, and M. Chen, "Extended state observer-based integral sliding mode control for an underwater robot with unknown disturbances and uncertain nonlinearities," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 8, pp. 6785–6795, 2017.

[22] M. D. Thekkedan, C. S. Chin, and W. L. Woo, "Virtual reality simulation of fuzzy-logic control during underwater dynamic positioning," *Journal of Marine Science and Application*, vol. 14, no. 1, pp. 14–24, 2015.

[23] C. S. Chin, W. P. Lin, and J. Y. Lin, "Experimental validation of open-frame rov model for virtual reality simulation and control," *Journal of Marine Science Technology*, vol. no. 2, p. 21, 2017.

[24] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[25] O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality motion planning," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 473–483, 2013.

[26] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[27] Y. Li, Z. Littlefield, and K. E. Bekris, "Sparse methods for efficient asymptotically optimal kinodynamic planning," *Springer Tracts in Advanced Robotics*, vol. 107, pp. 263–282, 2015.

[28] D. J. Webb and J. Van Den Berg, "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics," Computer Science, 2012.

[29] P. Pharpatara, B. Herisse, and Y. Bestaoui, "3-D trajectory planning of aerial vehicles using RRT*," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 1116–1123, 2017.

[30] D. Lee, H. Song, and D. H. Shim, "Optimal path planning based on spline-RRT* for fixed-wing UAVs operating in three-dimensional environments," in *Proceedings of the 14th International Conference on Control, Automation and Systems, ICCAS '14*, pp. 835–839, October 2014.

[31] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: a review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[32] V. Boor, M. H. Overmars, and A. F. V. D. Stappen, *Gaussian Sampling for Probabilistic Roadmap Planners*, Utrecht University Thenetherlands, Utrecht, Netherlands, 2001.

[33] S. Karaman and E. Frazzoli, "Sampling-based motion planning with deterministic μ-calculus specifications," in *Proceedings of the 48th IEEE Conference on Decision and Control*, pp. 2222–2229, December 2009.

[34] J. M. Esposito, J. Kim, and V. Kumar, "Adaptive RRTs for Validating Hybrid Robotic Control Systems," in *Algorithmic Foundations of Robotics VI*, vol. 17 of *Springer Tracts in Advanced Robotics*, pp. 107–121, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[35] S. M. LaValle and J. J. Kuffner Jr., "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[36] R. Motwani, *Randomized Algorithms*, Cambridge University Press, Cambridge, UK, 1995.