Covering space with simple robots: from chains to random trees

Asish Ghoshal and Dylan A. Shell

Abstract— **Inspired by the Rapidly-exploring random tree data-structure and algorithm for path planning, we introduce an approach for spanning physical space with a group of simple mobile robots. Emphasizing minimalism and using only InfraRed and contact sensors for communication, our position unaware robots physically embody elements of the tree. Although robots are fundamentally constrained in the spatial operations they may perform, we show that the approach — implemented on physical robots— remains consistent with the original data-structure idea. In particular, we show that a generalized form of Voronoi bias is present in the construction of the tree, and that such trees have an approximate space-filling property. We present an analysis of the physical system via sets of coupled stochastic equations: the first being the rate-equation for the transitions made by the robot controllers, and the second to capture the spatial process describing tree formation. We are able to provide an understanding of the control parameters in terms of a process mixing-time and show the dependence of the Voronoi bias on an interference parameter which grows as $O(\sqrt{N})$.**

I. Introduction

This paper considers the problem of having a group of simple mobile robots span a physical space. We describe an approach that is useful for simple robots, equipped only with limited sensing and short-range communication, in circumstances requiring systematic search or coverage of a space. Several researchers, starting with [1], but also [2] and [3], demonstrated algorithms for forming chains of robots. In these cases, a subset of the available robots move to locations that serve to guide peers, or maintain shared information, allowing the group to collectively overcome individual sensing limitations and improve performance, *e.g.,* in foraging, transport or delivery tasks.

We show how mobile robots can form an incremental tree by following a process analogous to a well known data-structure, the Rapidly-exploring random tree (RRT) [4]. In fact, "analogous" is too weak a word. The robots actually implement the tree creation algorithm, but they do in an unconventional way: information usually stored in programmatic variables is encoded in the robot's poses. This falls within the family of work on reducing, externalizing and redistributing information required for performing tasks, *cf.* [5, 1, 6]. In this paper, the robots store information in space by "being there."

Apart from the fact that trees generalize chains, this work was motivated by the particular strengths of the RRT: its ability to rapidly explore a configuration space through its bias towards unexplored regions, and its space filling property in the limit of many samples. These properties are desirable for groups of simple robots attempting to span a physical space, and we show that versions of these properties carry over to our implementation. For example, Voronoi bias, which causes the tree to grow rapidly towards larger unexplored regions, is preserved under conditions on the density of robots.

II. Related Work

The RRT was adopted because of the attractive properties of the process that produces the tree (*e.g.*, the ease with which the underlying operations can be translated into physical actions) and properties of the tree itself. As we have interpretations and treatments of some of those properties, we highlight important theoretical models and analysis of sampling based path planning algorithms here:

- Lamiraux and Laumond [7] were the first to study the probabilistic convergence of random sampling based planners using the theory of Markov chains and diffusion processes to analyze the RPP algorithm, concluding that the probability of failing to find a valid path when one exists decreases exponentially with the number of samples. Similar analysis was performed for RRTs [8], and, hence, its probabilistic completeness. Also, recently, non-optimality of RRT's paths was characterized [9].
- Kuffner and LaValle [10] studied RRTs as space-filling trees. In this paper we build upon our previous work [11] to define a notion of an (δ, ε)–space-filling tree and also formally analyze the Voronoi bias of the RRT. We also present a model for the expected contact distance for the points of the tree. Such contact distance distributions are a useful way to study simple point processes and thus provide important insights into embeddings of RRTs.

Additionally, we employ the macroscopic rate-equation model widely used to analyse the performance of the multi-robot swarms [12, 13, 14]. Important recent work has attempted to extend these models to capture time-delays [15], and treat spatial properties [16]. Herein, we couple a description of the distribution of controller states with a stochastic model of the tree and its growth, enabling some spatial dependency to be captured without resorting to partial differential equations.

III. Approach

A. The RRT data-structure

Algorithm 1 is the original algorithm for constructing an RRT for a general configuration space; here q_{init} is the initial configuration and G represents the tree.

Algorithm 1 BuildRRT

```
Require: $q_{\text{init}}$, $I$, $\Delta q$
1: $G.init(q_{\text{init}})$;
2: for $i = 1$ to $I$ do
3:     $q_{\text{rand}} \leftarrow$ RandConf();
4:     $q_{\text{near}} \leftarrow$ NearestVertex($q_{\text{rand}}$, $G$);
5:     $q_{\text{new}} \leftarrow$ NewConf($q_{\text{near}}$, $\Delta q$);
6:     $G.add\_vertex(q_{\text{new}})$;
7:     $G.add\_edge(q_{\text{near}}, q_{\text{new}})$;
8: end for
9: return G
```

A random configuration, q_{rand}, is chosen in each iteration which determines the new vertex that will be added to the graph G. The vertex in the tree, q_{near}, which is nearest to q_{rand} is computed. In step 5 of the algorithm a new configuration, q_{new}, is computed by selecting an action that moves q_{near} a distance, Δq, in the direction of q_{rand}. In the final step of each iteration, the vertex q_{new} and the corresponding edge $(q_{\text{new}}, q_{\text{near}})$ is added to the tree. In our implementation, robots represent both vertices and edges of G, where vertex and edge robots are identified based on the operations that they can perform. Fig. 1 shows vertex robots (colored in blue) and edge robots (colored in grey).Initially, all the robots except the robot denoting the root of the tree, start out wandering in the arena.The key difference between the above algorithm and ours is that we grow the tree asynchronously, via two operations:

ADDEDGE: a new edge is added to the tree that grows progressively towards q_{new}.

EXTENDEDGE: robots are added as edge nodes in order to extend the edge each time until the length of the edge becomes Δq. (Abusing notation slightly, we use $\Delta q \in \mathbb{Z}^+$ to denote the number of robots on an edge, unlike line 5 above.)

By asynchronous we mean that the steps of the algorithm are not atomic but span multiple iterations and may happen in parallel *e.g.,* while a new edge is being added on to the tree at one point (ADDEDGE operation), another robot might be extending an incomplete edge (EXTENDEDGE) so long as they do not interfere with each other. (Analysis below will model interference directly as a function of robot density.)

1) Initialization: The root of the tree, q_{init}, is logically denoted by placing a static robot in the environment. Other robots, which we call "wandering robots", perform a random walk to reach different points within the workspace space.

2) Choosing a random configuration: The random configuration, q_{rand}, is obtained the following way: wandering robots independently transition (with some probability) to a "spiralling" state wherein they execute the maneuver described in the next paragraph. Assuming robots perform random walks for sufficient time (*i.e.*, the probability of spontaneously transitioning is small so they randomly walk for a time comparable to the mixing-time) then the robot approximates a configuration chosen uniformly at random.

3) Finding the nearest neighbor: The vertex in the tree nearest to q_{rand} is found by having the robot spiral out until it bumps into a vertex robot which is already part of the tree. A spiraling robot may bump into another wandering robot in which case it resumes a random walk (*i.e.,* q_{rand} is discarded). If either spiraling or wandering robots bump into an edge robot then the robot begins to trace the tree in an attempt to complete an incomplete edge, *i.e.,* performing the EXTENDEDGE operation.

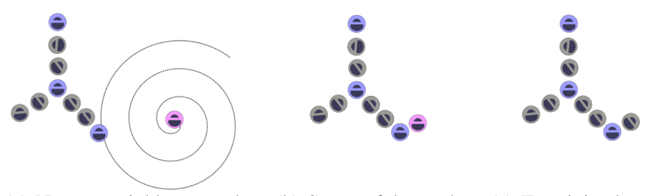
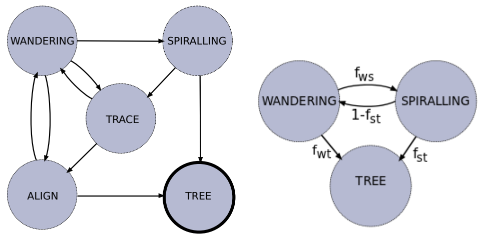

(a) Nearest neighbor search (b) Successful search (c) Tree joined

Fig. 1: Illustration of a spiralling robot joining the tree by initiating a new edge. The blue robots depict vertex robots while the grey robots denote edge robots. The edge length (Δq) is 2.



(a) Actual states (b) Simplified states

Fig. 2: Controller states of a robot.

4) The ADDEDGE *operation:* This operation is initiated only when a spiraling robot bumps into a fully formed edge (vertex robot) in which case it is added in place to the tree, increasing the edge in the direction of the randomly chosen point from where the robot started spiraling. If a spiraling robot bumps into an edge robot, it begins tracing the tree and if it finds a half formed edge then the EXTENDEDGE operation is initiated. This is shown in Fig. 1.

5) EXTENDEDGE *operation:* This operation occurs when wandering or spiraling robots bumps into an incomplete edge (some edge robot). The robot is added to the tree as an edge robot by aligning itself with the existing edge in straight line. While multiple EXTENDEDGE operations need to be performed to fill out the edge, they make up only a single "add_edge" operation in the original algorithm.

The abstracted controller states of a robot are shown in Fig. 2(a). The two ways in which a robot can join the tree becomes clear by looking at the state transitions of a single robot. To initiate a new edge a wandering robot transitions to the spiralling state and then to the tree state if possible. Upon failure to transition to the tree state from the spiralling state, a robot tries to join the tree by extending incomplete edges through a series of transitions through the *trace* and *align* states. A wandering robot may also join the tree by extending incomplete edges.

IV. IMPLEMENTATION

In order to prove the feasibility of the algorithm, we implemented the algorithm in our laboratory using eight iRobot Create robots, each equipped with an Asus Eee 1005HA netbook and augmented with an IR LED transmitter capable of transmitting one byte of information. A paper reflector was used to disperse the IR radiation with the aim of creating an IR field around the robot. The experimental arena was an $3.66\text{m} \times 5.05\text{m}$ rectangular area with the starting point located near the center of the top edge of the arena.

The first set of experiments aimed at forming a tree with all robots starting in wandering state, while the next set of experiments demonstrated four robots joining a partially formed tree of three robots (one vertex robot and two edge robots). Experiments were repeated by varying the parameter Δq from 1 to 3. We were successfully able to form trees of seven robots with Δq set to 3. (Fig. 3 is an example from such a trial.) This combination proved to be the most efficient in terms of the time taken for all robots to join the tree.

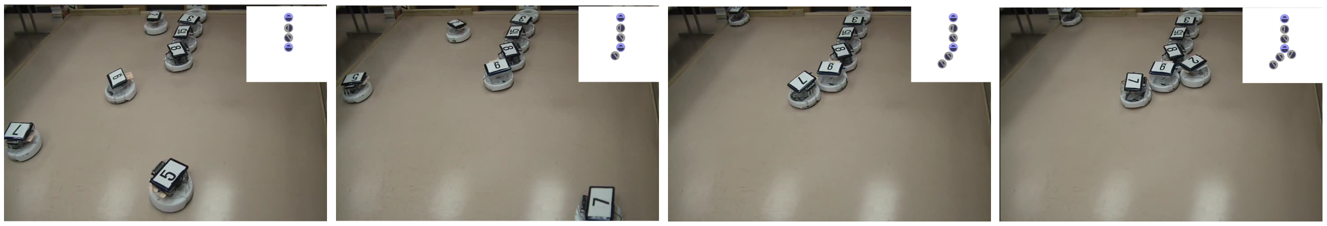


Fig. 3: A typical trial using 7 iRobot Create robots with $\Delta q = 3$. The logical tree is shown inset. A branch has been formed in the tree, but edges with robot #2 and robot #7 are not yet complete. Additional robots would need to perform EXTENDEDGE operations for that to occur. This illustrates the asynchronous growth process involved in forming the RRT.

A set of experiments were run where bumper taps did away with IR communication messages. A protocol was devised where different taps *viz.* short tap, long tap, double tap, *etc.* conveyed information. Success was had with Δq as 1, but alignment for the EXTENDEDGE operation proved challenging when $\Delta q > 1$ as communicative bumps slowly pushed robots out of the tree.

V. ANALYSIS

Motivated by the performance of the physical robot system, we developed a mathematical model of our algorithm in order to understand the parameters that have the most influence on the properties of the tree. In particular, we were concerned with the tree growth rate, bias of the tree growth processes, and the expected quality of the RRT (*i.e.,* space-filling behavior).

A. Definitions, Simplifications, and Roadmap

The stochastic process that generates the tree is influenced by the region in which the tree is embedded. We denote the subset of the two dimensional euclidean plane in which the tree is embedded by A. A critical factor is the sampling distribution, $\psi(x) : x \rightarrow [0,1]$ for all $x \subseteq A$, that determines the probability, $\mathbf{Pr}[q_{\text{rand}} \in x]$, with which the random samples are chosen from the bounded closed region x. The sampling distribution determines the Voronoi bias of the tree; it is desirable for it to be uniform over A so as to weight the exploration towards larger unexplored regions. So $\mathbf{Pr}[q_{\text{rand}} \in x] = \lambda_2(x)/\lambda_2(A)$, where $\lambda_2(.)$ is the area of a region.

A complication arising from the distributed construction of the data-structure is that multiple operations may proceed in parallel, without mapingp to a single atomic step in the original algorithm. Finding the nearest neighbor, computing the location of the new node, and finally adding the node to the tree are intermingled. There are several points of potential failure (*e.g.*, due to interference among robots, communication failure, *etc.*) from the time a point is *logically* sampled until the new node is added to the tree. To address this we introduce, and formalize in detail in § V-D, a joining distribution, which gives the probability of actually adding a node for each point sampled. As will be shown, this affects the performance of the system, the Voronoi bias, and the resulting tree.

Consider the following generalized model: the system consists of N homogeneous robots, executing identical controllers, in a convex polygonal obstacle free region A. The robots can be abstracted as points with radius of influence $\rho > 0$. Since in our case robots only employ contact sensors, the radius of influence is the radius of the robots. Our implementation only considered the case of A being a square, but the analysis holds for any convex region in the two dimensional euclidean plane. Convexity is imposed so that for $q_{\text{rand}} = x$, an edge can be constructed incrementally that joins q_{near} to $q_{\text{new}} = y \in A$, which need not always be possible for a non-convex region. A non-convex polygonal region has complex influences on the joining distribution which are difficult to characterize analytically. We first consider the case in which the edge length, $k \equiv \Delta q$, is 0 *i.e.* all robots are vertex robots. We then extend the model for the more general case of edge length greater than 0. The RRT step size is $r = 2(k+1)\rho$.

The simplified controller with high level behaviors is in Fig. 2(b). Each circle corresponds to a robot's state at any point in time. A wandering robot begins the nearest neighbor search procedure once it transitions to the spiralling state. The rate at which those transitions occur is given by system design parameter f_{ws}. We assumed that a spiralling robot making contact with the tree immediately becomes a node in the tree, *i.e.,* transitions to the tree state. This rate of transition is given by f_{st} and is dependent on the size and structure of the tree, and interference between spiralling and wandering robots. Since only spiralling robots join the tree it may be tempting to set f_{ws} as high as possible to maximize rate at which the tree grows. Doing so cause the uniform density and independence of the samples to be violated, since the robot does not have an opportunity to "lose" the history of the last spiralled transition. While setting f_{ws} to a very low value gives a desirable sampling distribution, the tree growth can be tedious. The effects of the design parameters are the topics of subsequent sections: First we describe the method to compute the minimum value f_{ws} that ensures that the sampling distribution is *close* to uniform. Next, we examine the tree's dynamics to compute f_{st}, and characterize the rate at which the tree grows given f_{st} and f_{ws}.

Although we use the standard rate equation approach developed in [13], what distinguishes this work is coupling between two different stochastic processes, one characterizing the random tree itself and the other characterizing the interactions between multiple agents to form the tree.

B. Uniform Sampling: Understanding f_{ws}

For the sampling of q_{rand} to be uniform, the positions from which robots start spiralling should be uniform over A. Let τ denote the time for which a wandering robot should wander before transitioning to the spiralling state. So τ must be *sufficient* to ensure independence. If the stochastic process X_t denotes the location of a wandering robot at time t then: $\mathbf{Pr}\left[X_{t+\tau} \in y | X_t \in x\right] = \lambda_2(y)/\lambda_2(A)$ for all bounded $x, y \subseteq A$. Thus, if the wandering robot is uniformly distributed at time t *i.e.,* $\mathbf{Pr}\left[X_t \in x\right] = \lambda_2(x)/\lambda_2(A)$, then $X_{t+\tau}$ and X_t would reflect independence. Since robots are initially (at time $t = 0$) distributed uniformly over the area A, a wandering robot should wander for $t \geq \tau$ before spiralling out so that the sampling distribution remains uniform. If the average speed of the wandering robot is ϑ then the distance covered by the wandering robot in time Δt is $\vartheta \Delta t$. The wandering motion of the robot can be approximated by dividing the region A into $(\vartheta \Delta t) \times (\vartheta \Delta t)$ sized grids and assuming that if at time t the robot is present at a particular cell, the position of the robot at time $t + \Delta t$ is uniformly distributed over the eight neighboring cells and the current cell. Thus, a wandering robot can be thought of as executing a random walk on a three dimensional torus with the number of vertices, $n = \left\lceil \sqrt{\lambda_2(A)/(\vartheta \Delta t)^2} \right\rceil$. Since the transition matrix of the random walk is symmetric, the stationary distribution is uniformly distributed over all the states. Thus, if τ_{mix} is the mixing time of the Markov chain then the parameter τ is given by $\tau = \tau_{mix} \Delta t$. The analysis is simplified if we consider the walk to be *lazy*, that is, at every time step the robot stays at its current position with probability $1/2$ and moves to one of its neighbors otherwise. Also the mixing time obtained for the lazy random walk can be used as an upper bound on the actual mixing time. The following theorem is due to [17].

Theorem 1: For the lazy random walk on the d-dimensional torus $\mathbb{Z}_n^d$,

$$\tau_{mix}(\varepsilon) \leq d^2 n^2 \log_2(\varepsilon^{-1}), \tag{1}$$

where ε is the variation distance (via stationary distribution). Thus, the parameter τ is given by:

$$\tau = \tau_{mix} \Delta t \leq \frac{9\lambda_2(A)}{4\rho^2} \log_2(\varepsilon^{-1}).$$

Where the last line follows from the fact that Δt is chosen such that $\vartheta \Delta t = 2\rho$, the diameter of the robot. Since there is at most N wandering robots at a given time out of which one should be selected to join the tree, the transition rate for each robot of transitioning from wandering state to spiralling state is given by, f_{ws}:

$$f_{ws} = \frac{1}{N\tau}, \tag{2}$$

which is a function of ε, selected based on how close to uniform we desire the sampling to be. Next, we model the stochastic process that generates the RRT.

C. Modelling Random Trees: Understanding f_{st}

The RRT vertices are a realization of a simple point process in $\mathbb{R}^2$; let T_n be the random set denoting the tree of n nodes, $n \in \mathbb{N}^*$, and let $p_n^m \in T_n$ denote the m^{th} point in the random set T_n where $m \in \mathbb{N}$ and $0 \leq m < n$. $D(p_n^m, r)$ is the closed disc of radius r centered at p_n^m and $C(p_n^m, r)$ is the circle of radius r centered at p_n^m, similarly $d(p_n^m, r)$ denotes the open disc. $T_1 = \{q_{\text{init}}\}$ where $q_{\text{init}} \in A$ is the tree's root.

Consider T_2; Let $T_2 = T_1 \bigcup \{x |$ for some $x \in A\}$. The next node, x, will lie in the closed disc $D(p_1^0, r)$ given the RRT step size r. Yet, the probability of the next point lying in the closed disc $D(p_1^0, r)$ is not uniform over the area of the disc *i.e.* $\mathbf{Pr}\left[x \in B | x \in D(p_1^0, r)\right] \neq \lambda_2(B)/\lambda_2\left(D(p_1^0, r)\right)$ for all $B \subseteq D(p_1^0, r)$, where B is a bounded closed set, since specifically, $\mathbf{Pr}\left[x \in d(p_1^0, r) | x \in D(p_1^0, r)\right] = \lambda_2\left(d(p_1^0, r)\right)/\lambda_2(A)$ and $\mathbf{Pr}\left[x \in C(p_1^0, r) | x \in D(p_1^0, r)\right] = 1 - \lambda_2\left(d(p_1^0, r)\right)/\lambda_2(A)$. Therefore, any tree can be thought of as comprised of long edges (with length r) and short edges (length $< r$). The probability of adding an edge depends on the type of edge. It is more complicated for T_n, when $n > 1$, since the probabilities depend on the area of Voronoi regions induced by points in the tree. Despite the distribution of points in the tree converging in probability to the sampling distribution [18], this does not provide information about the distribution of the points in the tree (T_n) for small values of n. Being unable, therefore, to reason about the distribution of realizations of RRTs, we consider the stochastic process C_n defined as:

$$C_n = \lambda_2\left(\bigcup_{i=0}^{n-1} D(p_n^i, r)\right), \tag{3}$$

i.e. C_n is the area of the union of all *r-discs* of the points in tree T_n. If the $(n+1)^{th}$ point lies outside area C_n then a long edge will be added to the tree, while if the point lies inside C_n then it will be added in place, producing a short edge. Characterizing the distribution of C_n does not determine the distribution of T_n, but for our purposes it suffices to compute $\mathbf{E}\left[C_n\right]$. Intuitively, C_n can be thought of as the area "covered" by the tree. In this way, we define what we call an (δ, ε)-*space-filling* tree.

Definition 1: A tree T_n for any $n \in \mathbb{N}^*$ is called (δ, ε)-*space-filling* in $[0, 1]^2$ if there exists a node p_n^i in the tree such that any point $x \in [0, 1]^2$ is within ε distance of p_n^i for $1 \leq i \leq n$ with probability at most δ for any $\varepsilon, \delta \in [0, 1]$.

Thus, for the tree T_n if $\mathbf{Pr}\left[C_n \geq c\right] \leq p$ for some $c \in \mathbb{R}$ and $c \leq \lambda_2(A)$, then the tree is (δ, r)-*space-filling* in A with $\delta = pc/\lambda_2(A)$ here, again, r is the RRT step size. The stochastic process C_n determines how quickly the distribution of points in the tree converge to the uniform distribution given a uniform sampling distribution. By using the Markov's inequality we get $\delta = \mathbf{E}\left[C_n\right]/\lambda_2(A)$.

In our physical robot implementation, the edges consist of physical agents and for a given edge length k, the tree can only be (δ, r)-space filling with $r = 2(k+1)\rho$ since the step size is fixed and all edges are of length r; some points within the region can only be r units close to a vertex since robot that start spiralling out a distance less than r from the tree do not join the tree.

To analyze tree coverage processes, we first consider the simplest case when A is the real line $[0, R]$ and the step size is r. Suppose that only long edges are added to the tree. When q_{rand} is less than r distance from its nearest neighbor, then it is not added to the tree, so $C_{n+1} = C_n$. The subscript n here denotes the number of points that have been sampled, which may exceed vertices in the tree $|T_m|$, because short edges are not added. For the one dimensional case we define C_n as:

$$C_n = \lambda_1\left(\bigcup_{i=0}^{m-1} D(p_m^i, r)\right), \tag{4}$$

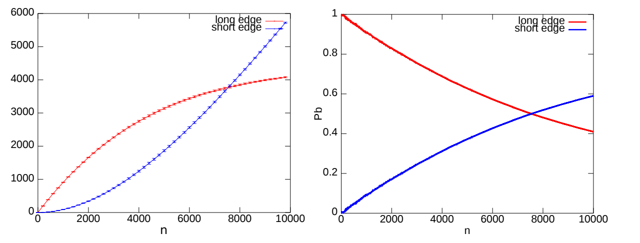


Fig. 4: Relative proportion of long edges in red vs. short edges in blue. (Left) Mean and Standard Deviation for number of long edges and short edges in a RRT as a function of number of nodes in the tree and (Right) the corresponding probabilities of getting a long edge and short edge for the experimental setup.

where $\lambda_1(.)$ represents the length of the line segment contained within the union of all $r-discs$. Thus, for the one dimensional case, C_n is a linear tree with a length that is a multiple of r. Introduce the function $p_l(n)$ defined as:

$$p_l(n) = \sum_{k=1}^{n-1} \mathbf{Pr}\left[C_n = (k+1)r | C_{n-1} = kr\right] \mathbf{Pr}\left[C_{n-1} = kr\right]. \quad (5)$$

Here, $p_l(n)$ denotes the probability that the n^{th} sampled point will lie on a long edge. Starting from (5) we get:

$$\begin{aligned} p_l(n) &= \sum_{k=1}^{n-1}\left(1-\frac{kr}{R}\right)\mathbf{Pr}\left[C_{n-1}=kr\right] && (6)\\ &= 1-\frac{1}{R}\sum_{k=1}^{n-1}(kr)\mathbf{Pr}\left[C_{n-1}=kr\right] = 1-\frac{\mathbf{E}\left[C_{n-1}\right]}{R}. && (7)\end{aligned}$$

Thus, $\mathbf{E}[C_n]$ is given as $\mathbf{E}[C_n] = (1-p_l(n+1))R$, which yields bounds on $p_l(n)$:

$$\begin{aligned} p_l(n) &\geq \mathbf{Pr}\left[L_n = nr\right] = \prod_{i=1}^{n-1}\left(1-\frac{ir}{R}\right)\\ &\geq \left(1-\frac{(n-1)r}{R}\right)^{(n-1)} \approx e^{-(n-1)^2 r/R} && (8)\end{aligned}$$

The general two dimensional case follows:

$$\mathbf{E}\left[C_n\right] = (1-p_l(n+1))\lambda_2(A). \quad (9)$$

In the general case C_n is a discrete time non-homogenous Markov chain on a continuous state space. We obtained $p_l()$ experimentally by computing the fraction of long edges that are present in a tree with n nodes. In general $p_l(n)$ decreases exponentially and for RRT of step size, $r=2$ and embedded in a 100×100 sqaure, $p_l(n)$ takes the form:

$$p_l(n) = \alpha e^{-\beta n}, \quad (10)$$

for some positive constants α and β. We obtained values for these parameters for a given scenario by performing simulations in CGAL [19]. Fig. 4 shows the number of short edges and long edges for an RRT and the corresponding probabilities of obtaining a type of edge; note the exponential curve. The experimentally determined parameters were $\alpha = 0.99$ and $\beta = -0.00009$.

D. Modelling Interaction Dynamics

Let Q_t be the stochastic process representing the state of a robot at time t with the state space given as $\{wandering, spiralling, tree\}$. The variable Q_t is macroscopic variable of the system— the fraction of robots in a particular state. Next, we examine the process Q_t describing the growth of the tree, and whose transition probabilities are determined by the stochastic process C_n.

The state of the system is given by three variables: $N_s(t), N_w(t)$, $N_r(t)$ where representing the number of spiralling, wandering, and tree robots, at time t with N being the total number of robots. The transition rate of a robot transitioning from wandering to the spiralling is given by f_{ws} as mentioned earlier and is obtained from (2). A spiralling robot transitions to the tree state only when it collides with a tree robot. If a spiralling robot starts spiralling out at a distance d from its nearest neighbor in the tree, then the probability of it transitioning to the tree state is given by the probability that the spiralling robot covers a radial distance d without colliding with a wanderer. A spiralling robot traces out an archimedean spiral given by $r = a + b\theta$, with the separation between successive rings, $2\pi b$, being constant. The radial distance covered by the robot is determined from the arc length of the spiral, given by $S(\theta) = (1/2)(b)\left[\theta\sqrt{1+\theta^2} + \ln(\theta + \sqrt{1+\theta^2})\right]$. We assume that the tangential acceleration of the spiralling robot, a, is constant. After differentiating $S(\theta)$ and some algebra, the radial distance covered up to time t is: $r(t) = \sqrt{(a\rho/\pi)}t^1$.

Now let B_n be the bounded closed region formed by union of all the $r-discs$ of the points in the tree T_n, that is,

$$B_n = \bigcup_{i=0}^{n-1} D(p_n^i, r). \quad (11)$$

By definition, $C_n = \lambda_2(B_n)$. At any time the wandering robots are distributed uniformly over the region A/B_n. The following lemma gives the expected distance of a randomly sampled point from its nearest neighbor among a set of points drawn from a distribution $f(x)$ over the unit square in $\mathbb{R}^2$.

Lemma 1: Let $f(x) > 0$ be a probability density function defined on the unit square $Q = [0,1]^2$ in $\mathbb{R}^2$. Let $U = \{X_1, X_2, \cdots, X_n\}$ be a set of n independent samples drawn drawn from $f(x)$. The expected contact distance $\mathbf{E}[D_n(x)]$ is given by:

$$\mathbf{E}\left[D_n(x)\right] = \frac{1}{2\sqrt{n}}\int_Q f(x)^{-1/2}dx \quad (12)$$

Proof: Let $\mathbf{Pr}[D_n(x) > k]$ be the probability that the contact distance is at least k. This means $X_i \notin D(x,k)$ for $1 \leq i \leq n$, *i.e.,* no point in the tree is present inside the disc of radius k centered at x. The expectation is given by[2]:

$$\begin{aligned}\mathbf{E}\left[D_n(x)\right] = \int_Q\int_0^\infty (1-U(x,k))^n dkdx &\approx \int_Q\int_0^\infty e^{-\pi n f(x)k^2}dkdx\\ &= \frac{1}{2\sqrt{n}}\int_Q f(x)^{-1/2}dx.\end{aligned}$$

Since, there are $N - N_r(t)$ robots distributed uniformly over the region $A/B_{N_r(t)}$, the mean free time of a robot follows from (12) and is then given by[3]:

$$\begin{aligned}\tau_{free}(N_r) &= \frac{1}{2v_{rel}\sqrt{N-N_r(t)}}\int_{A/B_{N_r(t)}}\frac{1}{\sqrt{\lambda_1(A)-\lambda_1(B_{N_r(t)})}}dx\\ &= \frac{1}{2v_{rel}}\sqrt{\frac{\lambda_1(A)-\lambda_1(B_{N_r(t)})}{N-N_r(t)}}. && (13)\end{aligned}$$

The average collision rate given by $1/\tau_{free}$ is constant for a given free space and number of tree robots at time t, since collisions between robots are independent events. The

[1]The radial distance covered depends on how the robots trace out their spirals. Our controller maintains constant angular velocity and increases the tangential velocity linearly.

[2]$\mathbf{Pr}[D_n(x) > k] = \int_Q (1-U(x,k))^n dx$ where $U(x,k) = \int_{D(x,k)} f(x)dx$. $U(x,k) \approx \pi f(x)k^2$, and $(1-U)^n \approx e^{-nU}$.

[3]v_{rel} is the average relative speed of a robot.

collision times can be then be approximated by a Poission distribution: the time between collisions is exponentially distributed with rate $\lambda_t = 1/\tau_{free}(N_r)$. The random variable S represents the time for which a spiralling robot spirals out before coming in contact with a wandering robot, thus, has the following distribution:

$$\mathbf{Pr}[S \leq t] = \begin{cases} 1 - e^{-\lambda_t t}, & t \geq 0 \\ 0, & t < 0. \end{cases} \tag{14}$$

Now, consider the Voronoi tessellation of the region A so that V_k represents the Voronoi cell of the node $k \in T_n$, *i.e.,* $V_k = \{x \mid ||x,k|| \leq ||x,j||, \forall x \in A \text{ and } j \in A \text{ and } j \neq k\}$. If $V = \bigcup_{k \in T_n} V_k$, then $V(t)$ is the Voronoi region of the entire tree at time t. Then any robot spiralling out from within $A/V(t)$ would strike the boundary before any tree nodes, and would never join the tree. Note that $V(t)$ varies with time as the size of the tree grows. Thus, the nearest neighbor search is successful only if a robot starts from within region $V(t)$ and if it hits a tree robot without colliding with any wanderers. Let the probability that a robot starts spiralling out from within $V(t)$ be represented by $p_v(t) = \lambda_2(V(t))/\lambda_2(A)$. We can evaluate the joining distribution, $J(x)$, introduced earlier, as a function of distance from tree; *i.e.* $J(x)$ represents the probability of a spiralling robot, at a distance of at least x from the tree, will join the tree follows from memorylessness of the exponential distribution:

$$\begin{aligned} J(x) &= p_v(t)\mathbf{Pr}\left[Q_{t+r^{-1}(x)} = \text{tree} | Q_t = \text{spiralling}\right] \\ &= p_v(t)\mathbf{Pr}\left[S > r^{-1}(x)\right] = p_v(t) e^{-\lambda_t \sqrt{\frac{\pi}{a\rho}} x}. \end{aligned} \tag{15}$$

We simplify the analysis by ignoring boundaries, but effects of walls on performance is examined in the discussion section. Doing so, the simplified cumulative distribution function of $J(x)$ and associated probability density function $j(x)$ are:

$$\widehat{J}(x) = 1 - e^{-\lambda_t \sqrt{\frac{\pi}{a\rho}} x}, j(x) = \frac{d}{dx}(\widehat{J}(x)) = \lambda_t \sqrt{\frac{\pi}{a\rho}} e^{-\lambda_t \sqrt{\frac{\pi}{a\rho}}}. \tag{16}$$

Thus, the instantaneous transition rate of transitioning from the spiralling state to the tree state,f_{st}, is given by [4]:

$$\begin{aligned} f_{st} &= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \mathbf{Pr}[Q_{t+\Delta t} = \text{tree} | Q_t = \text{spiralling}] \\ &= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \left[\int_0^{\gamma \Delta t} \int_0^{2\pi} j(x) dx d\theta\right] \\ &= 2\pi\lambda_t = 4\pi v_{rel} \sqrt{\frac{N - N_r(t)}{\lambda_2(A/B_{N_r})}}. \end{aligned} \tag{17}$$

These rate equations describe the average number of wandering robots, spiralling robots and tree robots respectively with time:

$$\begin{aligned} \frac{d}{dt}\left(\widetilde{N}_w(t)\right) &= \widetilde{N}_s(t)(1 - f_{st}) - \widetilde{N}_w(t) f_{ws}, \\ \frac{d}{dt}\left(\widetilde{N}_s(t)\right) &= \widetilde{N}_w(t) f_{ws} - \widetilde{N}_s(t) f_{st} - \widetilde{N}_s(t)(1 - f_{st}), \\ \frac{d}{dt}\left(\widetilde{N}_r(t)\right) &= \widetilde{N}_s(t) f_{st}. \end{aligned} \tag{18}$$

[4] $\gamma = \sqrt{a\rho/\pi}$. The simplification on the first line reflects the fact that only spiralling robots in the $(\gamma\Delta t)$-disc around the nearest neighbor can reach the tree by time $t + \Delta t$. The last line shows that the transition rate depends on collision parameter λ_t.

The preceding analysis does not consider the time (and robots) involved in EXTENDEDGE operations. When $k > 0$, wanderers can join the tree to extend an incomplete edge. The state transitions must be augmented slightly. The probability of a wandering robot joining the tree is maximal when the length of the edge is $k+1$, *i.e.,* only one robot is required to complete the edge, since the wandering robot can hit any one of the $k+1$ robots, after which it will skirt the edge and join at the end. On the other hand, the probability of a wandering robot of joining the tree is 0 when the nearest neighbor is a vertex robot and no edge has been initiated. There are at most $\lceil N_r/k+1 \rceil$ vertex robots which can have edges $k+1$ robots long and the probability of completing one of those edges is $(k+1)/N_r$. Thus, the transition rate of transitioning from the wandering state to the tree state is given as follows:

$$f_{wt} \quad \leq \quad \frac{(k+1)}{\tau_{free}(N_r)} = \left(\frac{k+1}{N_r}\right) \frac{2 v_{rel} \sqrt{N - N_r}}{\lambda_2(A/B_{N_r})}. \tag{19}$$

Since a spiralling robot can initiate an edge and join the tree only when it hits a vertex robot and the probability of hitting a vertex robot is $1/(k+1)$, the modified transition probability of transitioning from the spiralling state to the tree state is given by: ${f_{st}}' = f_{st}/(k+1)$ and the rate equations for the general case of $k \geq 0$ is given by:

$$\frac{d}{dt}\left(\widetilde{N}_w(t)\right) = \widetilde{N}_s(t)(1 - f'_{st}) - \widetilde{N}_w(t).f_{ws} - \widetilde{N}_w(t) f_{wt}, \tag{20}$$

$$\frac{d}{dt}\left(\widetilde{N}_s(t)\right) = \widetilde{N}_w(t) f_{ws} - \widetilde{N}_s(t) {f_{st}}' - \widetilde{N}_s(t)(1 - {f_{st}}'), \tag{21}$$

$$\frac{d}{dt}\left(\widetilde{N}_r(t)\right) = \widetilde{N}_s(t) f'_{st} + \widetilde{N}_w(t) f_{wt}. \tag{22}$$

When $k > 0$, expression (18) gives a lower estimate of the average number of tree robots at time t, while (22) is an overestimate of the average number of tree robots.

E. Voronoi Bias

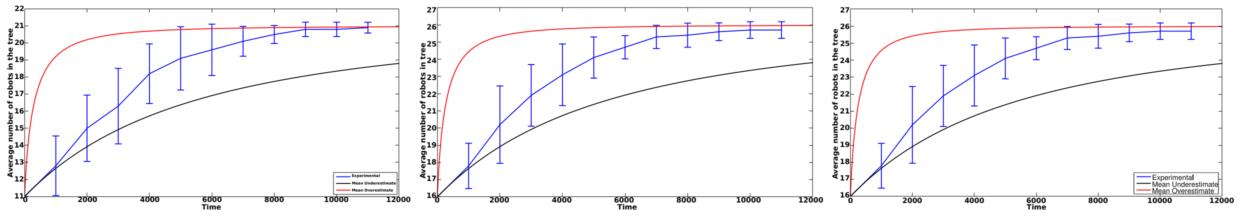
Earlier we described the *Voronoi bias* as a desirable property for the tree growth process, which we interpreted as implying an aspiration for uniformity in the sample distribution. Here we examine the effect of the joining distribution, $j(x)$, on the bias, broadening the notion of a Voronoi bias.

Definition 2 (Strict Bias): Let $P(x)$ be the parent of $x = T_{n+1}/T_n$, the $(n+1)^{th}$ node added to the tree, and let $G(k) : k \to [0,1]$ for all $k \in T_n$ such that $\sum_{k \in T_n} G(k) = p$, $\mathbf{Pr}[P(x) = k] = G(k)$ and $\mathbf{Pr}[x = \phi] = 1 - p$ for $p \in [0,1]$. Then the tree T_n is said to have a Voronoi bias if $\lambda_2(V(k))/\lambda_2(V(k')) = G(k)/G(k')$ for all $k, k' \in T_n$.

Distribution $G(x)$ sums to p over all the nodes in the tree which may be ≤ 1 (accounting for the fact that with positive probability none of the nodes may be picked as the nearest neighbor, giving $T_{n+1} = T_n$). From (16) we have $G(k)$ as:

$$G(k) = \int_{V_k} j(x) dx. \tag{23}$$

Thus, for a tree to have Voronoi bias the function $j(x)$ should be uniform over the region A *i.e.* $j(x) = p/\lambda_2(A)$; while, in our implementation, this is true as $\lambda_t \to 0$, it will be violated otherwise since $j(x)$ drops off exponentially with distance from the tree. Nevertheless, the following definition,



(a) k=1 (b) k=2 (c) k=2, with boundary

Fig. 5: Plot showing the underestimate (black) and overestimate (red) of the average number of tree robots at a given time. The average number of robots in the tree calculated experimentally are shown in blue.

in which nodes with larger Voronoi regions are favored over nodes with smaller regions (although not necessarily proportionally), still captures the intuitive idea of a exploration oriented bias.

Definition 3 (Weak Bias): Let $P(x)$ be the parent of $x = T_{n+1}/T_n$, the $(n+1)^{th}$ node added to the tree, and let $G(k) : k \rightarrow [0,1]$ for all $k \in T_n$ such that $\sum_{k \in T_n} G(k) = p$, $\mathbf{Pr}[P(x) = k] = G(k)$ and $\mathbf{Pr}[x = \phi] = 1 - p$ for $p \in [0,1]$. Then the tree T_n is said to have *weak* Voronoi bias if $\lambda_2(V(k)) \geq \lambda_2(V(k')) \Rightarrow G(k) \geq G(k')$ for all $k, k' \in T_n$.

In general, the joining distribution $j(x)$ depends not only on the area of $V(k)$ but also on its shape. While one can contrive cells in which this weak bias is violated (long, skinny cells having large area but limited $j(x)$ density) assessing their likelihood of occurrence would depend on an understanding of how "well-behaved" the distribution of Voronoi cells is. It is clear that this weaker definition depends on λ_t less strongly. Under either definition, it is clear is that the quality of the tree's expansion decreases with large λ_t, itself scaling as $O(\sqrt{N})$.

VI. Results

In order to carry out larger scale experiments to validate our model, we implemented a version of the algorithm in the Stage simulator. Experiments were performed with $N = \{15, 30\}$ robots in a square arena of size 15m×15m. The detection region was the radius of the robots, $\rho = 0.1778$m. Parameters were determined as: $\alpha = 0.99$, $\beta = 0.003393$, and $\gamma = 0.015$, the average relative velocity used was 0.07 and parameter f_{ws} was set to 0.01 sec^{-1}. Experiments were carried out with edge lengths $k = \{1, 2, 3\}$. Fig. 5 shows results averaged over 10 trials for each value of k. For $k = 1$ the initial size of the tree was 11, for $k = 2$ the initial size of the tree was 16.The initial size were chosen such that all trees had 6 nodes (vertex robots) including the initial node.

VII. Conclusion

The previous section compares results from simulation experiments and the overestimate and underestimate of the mean, finding good agreement between the model and experiment. The analysis relies on a few simplifications regarding the environment: the region is convex and obstacle free, and spiralling robots do not collide with the boundary. Boundary effects are taken into account(Fig. 5) by scaling f_{st} appropriately. Analysis for non-convex regions is complicated due to the fact that non-convex regions and obstacles introduce *shadows* and *holes* in the region which have complex influences on the joining distribution $j(x)$.

The work demonstrates what we believe to be a broader idea of *physical data-structures*: namely that several existing spatial algorithms with well-understood properties can be directly implemented on robot hardware so that the resulting properties describe the robots' configurations. The primitives employed by the algorithm point to behaviors that robots need to be able to execute. If an asynchronous implementation of a synchronous data-structure can be made consistent, then the algorithmic analysis can be carried over to the state of the robots themselves. In this particular case, we have extended and broadened some existing definitions (*e.g.,* distance dependant sampling success, Voronoi bias, and the space-filling property) in analyzing the behavior of our multi-robot system.

References

[1] B. B. Werger and M. J. Matarić, "Robotic 'Food' Chains: Externalization of State and Program for Minimal-Agent Foraging," in *Proc. Sim. of Adaptive Behr*, 1996, pp. 625–634.

[2] S. Nouyan, A. Campo, and M. Dorigo, "Path formation in a robot swarm," *Swarm Intelligence 2(1), pp.1–23*, 2008.

[3] F. Ducatelle, G. A. Di Caro, C. Pinciroli, and L. M. Gambardella, "Self-organized Cooperation between Robotic Swarms," *Swarm Intelligence Journal*, vol. 5, pp. 73–96, 2011.

[4] S. M. LaValle and J. Kuffner, Jr., "Rapidly-exploring random trees: Progress and prospects," in *Proc. Workshop on the Algorithmic Foundations of Robotics*, 2000.

[5] B. R. Donald, J. Jennings, and D. Rus, "Minimalism + Distribution = Supermodularity," *Journal of Exp. and Theoretical Artificial Intelligence*, vol. 9, no. 2–3, pp. 293–321, Apr. 1997.

[6] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee, "Pheromone Robotics," *Auto. Robots 11(3):319–324*, 2001.

[7] F. Lamiraux and J. P. Laumond, "On the expected complexity of random path planning," in *ICRA*, 1996, pp. 3306–3311.

[8] S. M. LaValle and J. J. Kuffner, Jr., "Randomized kinodynamic planning," *IJRR*, vol. 20, no. 5, pp. 378–400, 2001.

[9] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *ArXiv e-prints*, May 2011.

[10] J. J. Kuffner Jr. and S. M. LaValle, "Space-Filling Trees," RI, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-09-47, 2009.

[11] A. Ghoshal and D. Shell, "Being there, being the RRT: Space filling and searching with minimalist robots." in *AAAI Spr. Symp., Multirobot Systems and Physical Structures*, 2011.

[12] K. Sugawara and M. Sano, "Cooperative acceleration of task performance: Foraging behavior of interacting multi-robots system," *Physica D, 100(3/4), pp. 343–354*, 1997.

[13] K. Lerman and A. Galstyan, "A General Methodology for Mathematical Analysis of Multi-Agent Systems," USC Information Sciences Institute, Tech. Rep. ISI-TR-529, 2001.

[14] A. Martinoli, K. I. Easton, and W. Agassounon, "Modeling of Swarm Robotic Systems: A Case Study in Collaborative Distributed Manipulation," *IJRR, 23(4), pp. 415–436*, 2004.

[15] T. W. Mather and M. A. Hsieh, "Analysis of Stochastic Deployment Policies with Time Delays for Robot Ensembles," *IJRR*, vol. 30, no. 5, pp. 590–600, Apr. 2011.

[16] S. Berman, V. Kumar, and R. Nagpal, "Design of Control Policies for Spatially Inhomogeneous Robot Swarms with Application to Commercial Pollination," in *ICRA*, May 2011.

[17] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*. AMS, 2006.

[18] J. J. Kuffner Jr. and S. M. Lavalle, "Rrt-connect: An efficient approach to single-query path planning," in *ICRA*, 2000.

[19] "CGAL, Computational Geometry Algorithms Library," http://www.cgal.org.