RAPIDLY-EXPLORING RANDOM TREE INSPIRED MULTI-ROBOT SPACE COVERAGE

A Thesis

by

ASISH GHOSHAL

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2012

Major Subject: Computer Science

RAPIDLY-EXPLORING RANDOM TREE INSPIRED MULTI-ROBOT SPACE COVERAGE

A Thesis

by

ASISH GHOSHAL

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Dylan A. Shell
Committee Members,	Nancy M. Amato
	Richard J. Malak Jr.
Head of Department,	Duncan M. Walker

May 2012

Major Subject: Computer Science

ABSTRACT

Rapidly-exploring Random Tree Inspired

Multi-robot Space Coverage. (May 2012)

Asish Ghoshal, B.Tech., National Institute of Science and Technology

Chair of Advisory Committee: Dr. Dylan A. Shell

Inspired by the Rapidly-exploring Random Tree (RRT) 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 also introduce a class of fixed edge length RRTs called ℓRRT and show that ℓRRTs have similar space-filling properties to that of RRTs. 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 $O(\sqrt{N})$.

To Maa, Baba and Dida.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor Prof. Dylan Shell who has been a constant source of support and knowledge, for all his efforts in helping me write papers and this thesis and above all for being a great mentor and inspiring me to do good research.

I would also like to thank my fellow lab mates for helping me in preparing for me defense.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

FIGURE Page

CHAPTER 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 Werger and Matarić [1], but also Nouyan et al. [2] and Ducatelle et al. [3] more recently, developed and demonstrated algorithms for forming chains of mobile robots. In these cases, a subset of the available robots move to locations that serve to guide peer robots, or to maintain shared information, allowing the group to collectively overcome individual sensing limitations and improve performance, *e.g.,* in foraging, transport or delivery tasks. The basic idea is that some agents are most effective when forming and maintaining dynamic infrastructure for use by the group. Motivated by this same idea, we describe a algorithm for forming tree structures in physical space rather than the usual linear chains.

Within this paper, 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 probably 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 poses of the robots themselves. Some early influential multi-robot researchers investigated techniques for reducing, externalizing, redistributing information required for performing tasks, *cf.* [5, 1, 6]. That work showed how information deposited in the world can

---

´The journal model is *IEEE Transactions on Automatic Control.*

be effectively indexed and exploited via local spatial queries (*i.e.,* situatedness). In this paper, the robots store information in space by "being there"; the type of information is easily identifiable because it is more usually associated with an explicit data-structure.

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. After presenting a description of the approach (see chapter III), we analyze general RRTs and introduce a special type of RRT called ℓRRT (see chapter IV) which is followed by implementation (see chapter V) and an analysis of the multi-robot system (see chapter VI) to determine system performance and characterize various properties of the generated tree which is followed by experimental validation of the model (see chapter VII) and a brief discussion (see chapter VIII) before finally concluding (see chapter IX).

CHAPTER 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. However, the authors also recognize that the convergence rate is expressed with parameters that are difficult to compute for a given example; computing the rate in terms of more amenable parameters remains a problem. Also, recently, non-optimality of RRT's paths was characterized [9].

- Kuffner and LaValle [10] studied RRTs as space-filling trees. We 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 [11, 12, 13, 14]. Important recent work has attempted to extend these models to capture time-delays [15], and treat spatial

properties [16]. In our approach, 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.

CHAPTER III

APPROACH

A. The RRT data-structure and algorithm

Algorithm 1 is is original algorithm for constructing an RRT [17] for a general configuration space; here q_{init} is the initial configuration and G represents the tree.

Algorithm 1 BUILDRRT

Require: q_{init}, K, Δq

1: $G.init(q_{\text{init}})$;
2: **for** $k = 1$ to K **do**
3: $q_{\text{rand}} \leftarrow$ RANDCONF();
4: $q_{\text{near}} \leftarrow$ NEARESTVERTEX(q_{rand}, G);
5: $q_{\text{new}} \leftarrow$ NEWCONF(q_{near}, Δq);
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 per-

form. Fig. 1 shows vertex robots (colored in blue) and edge robots (colored in grey). 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. (We abuse the notation slightly, using $\Delta q \in \mathbb{Z}^+$ to denote number of robots on an edge, unlike line 5 above.)

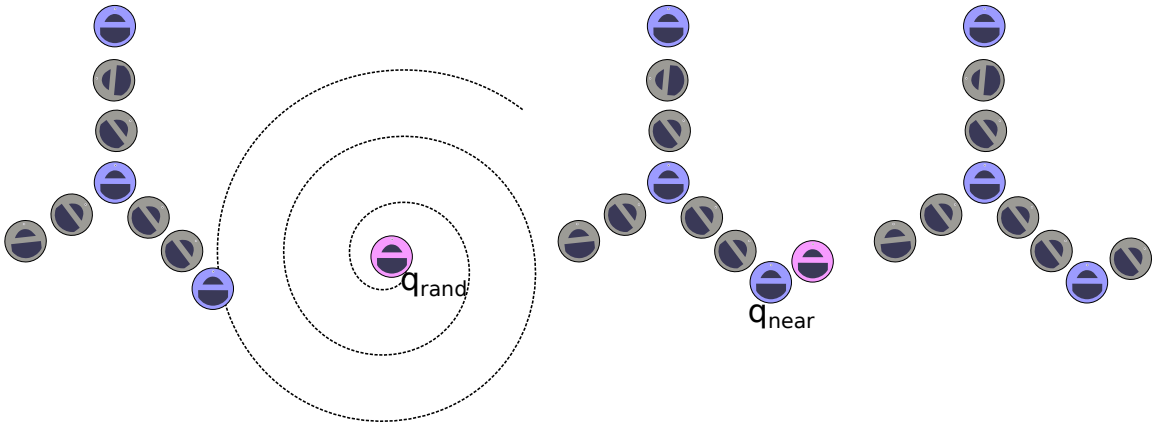
By asynchronous we mean that the steps of the algorithm are not atomic but span multiple iterations and may happen in parallel *e.g.,* while a new edge is being added on to the tree at one point (AddEdge operation), another robot might be extending an incomplete edge (ExtendEdge) so long as they do not 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 selected 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 "spiraller" state wherein they



(a) Nearest neighbor search

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.

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 enough that 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.

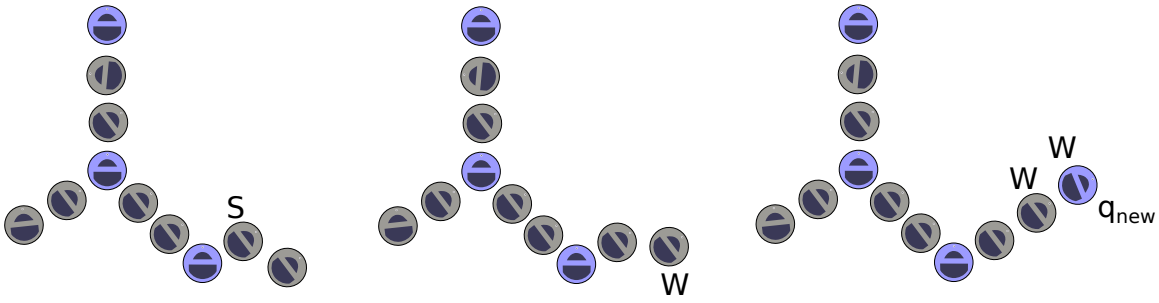


Fig. 2. An illustration of a wandering robot (W) extending an edge, initiated by a previously spiralling robot (S), by progressively aligning itself.

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 then traces the tree to find an incomplete edge. The robot is added to the tree as an edge robot (or a vertex robot if the robot completes the edge) by aligning itself with the existing edge in a straight line. Fig. 2 illustrates this. While multiple ExtendEdge operations need be performed to fill out the edge, they make up only a single "add_edge" operation in the original algorithm

Fig. 3 shows the controller logic of a single robot. Each colored block represents a state of the robot comprising of some collection of behaviors and actions that a

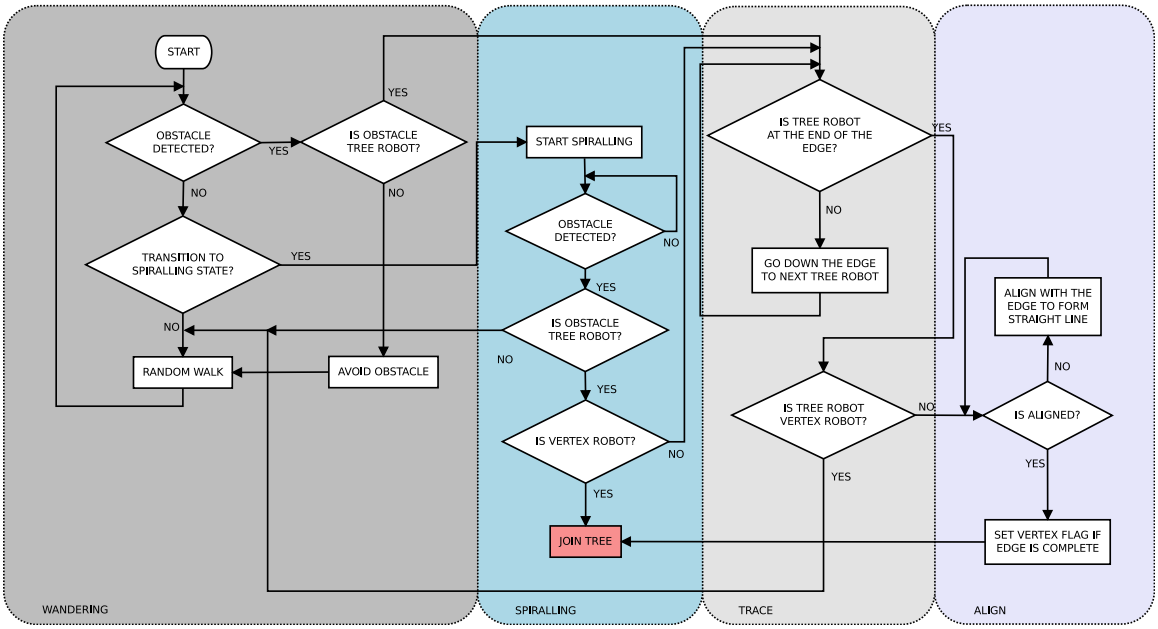


Fig. 3. Controller logic of a robot.

robot is executing. The simplified state diagram of a robot controller is shown in Fig. 4. A robot can join the tree in two ways, it can either join the tree by extending an edge in which case it progressively aligns itself along the existing edge or by initiating a new edge.

B. Relationship of the physical tree to the RRT: Introducing ℓRRT

In the previous section we presented the original RRT algorithm and the controller logic of the robots that form the physical tree. What differentiates our approach to building the tree from the traditional algorithm is that it is asynchronous and the RRT step size is discrete and is specified by the number of edge robots k. Thus the tree is grown asynchronously in steps of 2ρ, where ρ is the radius of a robot. A side-effect of the asynchronous approach is that there can be incomplete edges, *i.e.* edges that have length $2(k'+1)\rho$ with $k' < k$, at any given time. Further, the

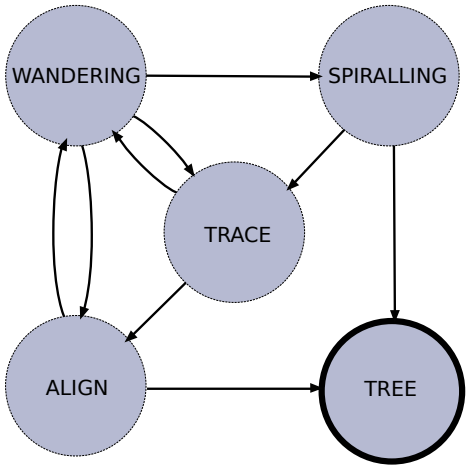


Fig. 4. Controller states of a robot.

edges might never complete because the robots have no global information about the structure of the tree and the presence of other incomplete edges. So we might end up with multiple incomplete edges at various points of the tree which could have been combined to form complete edges. Although steps can be taken to ensure complete edges: for instance, robots which are part of an incomplete edge can break off after some period of inactivity; however, there is no particular benefit of having complete edges considering the time it might take to complete edges. Rather, having robots at the end of each edge flip their *type* to vertex robots after some period of inactivity would ensure a tree that is logically complete with all internal nodes connected to their parents by edges of length exactly $2(k+1)\rho$ and the leaf nodes connected by edges of length at most $2(k+1)\rho$.

Another difference between the physical tree and the RRT is that points that are sampled at a distance less than the RRT step size from the tree are not added to the tree i.e. if a spiralling robot spirals out from a distance less than $2(k+1)\rho$, then it is not added to the tree. This is done so as to prevent edges from crossing each other since wandering robots always try to extend an edge until the edge length

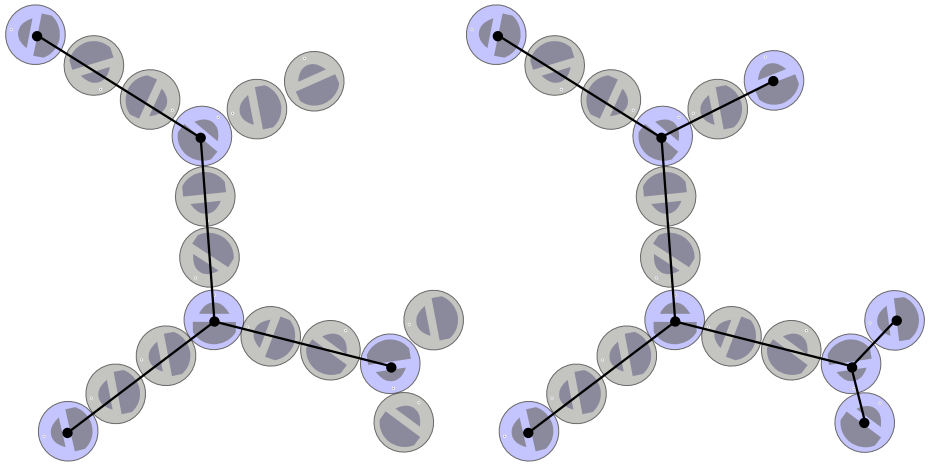


Fig. 5. Converting a tree with incomplete edges to a complete tree by flipping edge robots at the end of each edge to vertex robots after a period of inactivity.

is $2(k+1)\rho$. So the tree generated using such a process can be likened to a RRT in which the sampling distribution is such that points sampled closed to the tree are discarded and the resulting tree has only long edges *i.e.* edges of length exactly r, where r is the RRT step size. We call such RRTs ℓRRT. We show in section IV that RRTs formed using such a distribution are as good as general RRTs in filling a space.

In spite of these differences we believe that the process of generating the physical tree and the resulting tree is consistent with the conventional RRT algorithm and data-structure.

CHAPTER IV

ANALYSIS OF RRT AND *ℓRRT*

A. Analysis of general RRTs

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 root of the tree and is given.

Consider T_2; Let $T_2 = T_1 \bigcup \{x| \text{ for some } x \in A\}$. 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.* $\mathbf{Pr}\left[x \in B | x \in D(p_1^0, r)\right] \neq \lambda_2(B)/\lambda_2(D(p_1^0, r))$ for all $B \subseteq D(p_1^0, r)$, where B is a bounded closed set, since specifically, $\mathbf{Pr}\left[x \in d(p_1^0, r) | x \in D(p_1^0, r)\right] = \lambda_2(d(p_1^0, r))/\lambda_2(A)$, $\mathbf{Pr}\left[x \in C(p_1^0, r) | x \in D(p_1^0, r)\right] = 1 - \lambda_2(d(p_1^0, r))/\lambda_2(A)$. Therefore, any tree can be thought of as comprised of long edges (with length r) and short edges (length $< r$). The probability of being added depends on the type of edge. It is more complicated when T_n, for $n > 1$, 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 follows:

$$C_n = \lambda_2 \left(\bigcup_{i=0}^{n-1} D(p_n^i, r) \right), \tag{4.1}$$

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 yet knowing $\mathbf{E}[C_n]$ can give us useful information about a tree, like how space-filling a tree is, as we will show shortly. Intuitively, C_n can be thought of as the area "covered" by 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:

$$C_n = \lambda_1 \left(\bigcup_{i=0}^{m-1} D(p_m^i, r) \right), \tag{4.2}$$

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. We introduce the function $p_l(n)$ defined as follows:

$$p_l(n) = \sum_{k=1}^{n-1} \mathbf{Pr}\left[C_n = (k+1)r | C_{n-1} = kr\right] \mathbf{Pr}\left[C_{n-1} = kr\right]. \tag{4.3}$$

Here, $p_l(n)$ denotes the probability that the n^{th} sampled point will lie on a long edge. Starting from (4.3) we get the following equation:

$$\begin{aligned} p_l(n) &= \sum_{k=1}^{n-1} \left(1 - \frac{kr}{R}\right) \mathbf{Pr}\left[C_{n-1} = kr\right] \\ &= 1 - \frac{1}{R} \sum_{k=1}^{n-1} (kr) \mathbf{Pr}\left[C_{n-1} = kr\right] \\ &= 1 - \frac{\mathbf{E}\left[C_{n-1}\right]}{R}. \end{aligned} \tag{4.4}$$

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

$$\begin{aligned} p_l(n) &\geq \mathbf{Pr}\left[C_n = nr\right] \\ &= \prod_{i=1}^{n-1} \left(1 - \frac{ir}{R}\right) \\ &\geq \left(1 - \frac{(n-1)r}{R}\right)^{(n-1)} \\ &\approx e^{-(n-1)^2 r/R} \end{aligned} \tag{4.5}$$

The general two dimensional case follows:

$$\mathbf{E}\left[C_n\right] = (1 - p_l(n+1))\lambda_2\left(A\right) \tag{4.6}$$

In the general case C_n is a discrete time non-homogenous Markov chain on a continuous state space. $p_l(n)$ can be obtained experimentally by computing the frac-

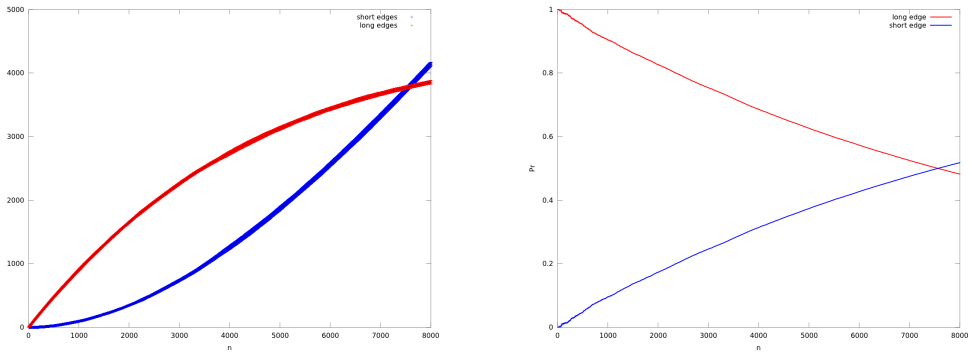


Fig. 6. Relative proportion of long edges in red vs. short edges in blue (left) for $r = 2$ in a square of length 100. Mean and Standard Deviation for number of long edges and short edges in a RRT as a function of number of nodes in the tree and (right) the corresponding probabilities of getting a long edge and short edge.

tion 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, $r = 2$, embedded in a sqaure of length 100 $p_l(n)$ takes the form:

$$p_l(n) = \alpha e^{-\beta n}, \tag{4.7}$$

for some positive constants α and β. We obtained values for these parameters for a given scenario by performing simulations in CGAL [19]. Fig. 6 shows the number of short edges and long edges for a RRT and the corresponding probabilities of obtaining a type of edge; note the exponential curve. The experimentally determined parameters were $\alpha = 0.99$ and $\beta = -0.00009$. Fig. 7 shows $p_l(n)$ for various RRTs embedded in a square of length 100.

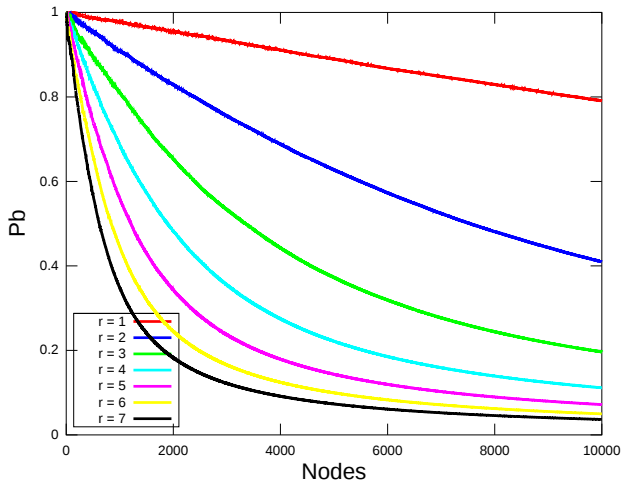


Fig. 7. Probability of getting long edge for RRT of edge length r generated in a square of length 100.

1. Space-filling property

In this section we characterize the space-filling property of RRTs by defining what we call as a (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 $c \leq \lambda_2\left(A\right)$, then the tree is (δ, r)-*space-filling* in A with $\delta = pc/\lambda_2\left(A\right)$ 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. We can obtain δ by using the Markov's inequality as follows:

$$\mathbf{Pr}\left[C_n \geq c\right] \leq \frac{\mathbf{E}\left[C_n\right]}{c}, \delta = \frac{\mathbf{E}\left[C_n\right]}{A} \tag{4.8}$$

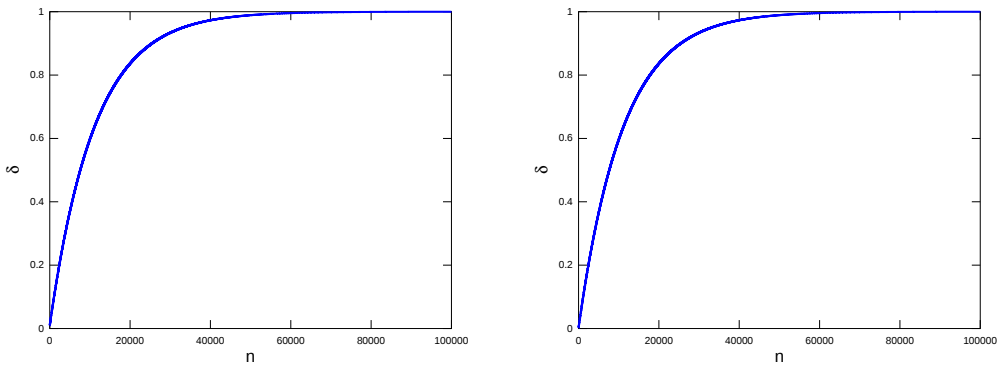


Fig. 8. δ for RRT(left) and ℓRRT(right) of edge length $r = 2$ generated in a square of length 100. It is clear that at 60000 nodes the tree has completely filled the space in the sense that every point is at most r distance away from a point of the tree in both the cases.

Fig. 8 shows δ for a RRT of step size 2 in a square of length 100.

2. Voronoi bias

The attractiveness of RRTs arises from the fact that the tree rapidly grows towards unexplored regions *i.e.* the growth of the tree is biased towards larger voronoi regions. To formalize that notion consider the voronoi tessellation of a region, $A \in \mathbb{R}^2$, introduced by the points in the tree so that V_k represents the voronoi cell of the node $k \in T_n$, *i.e.*,

$$V_k = \{x \mid ||x,k|| \leq ||x,j|| \,, \forall x \in A \text{ and } j \in A \text{ and } j \neq k\}. \tag{4.9}$$

. The bias of the tree is defined as follows:

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 $T_{n+1} = T_n$).

If the sampling probability density is $\psi(x)$ then $G(k)$ is given as follows:

$$G(k) = \int_{V_k} \psi(x)dx. \tag{4.10}$$

Thus, for a tree to have Voronoi bias the function $\psi(x)$ should be uniform over the region A, *i.e.* $\psi(x) = p/\lambda_2(A)$, to satisfy the definition of strict voroni bias; yet there might still be some bias for non-uniform sampling density functions. Next we present a weaker definition of Voronoi 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 $G(k) : k \to [0,1]$ 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* weak *Voronoi bias if $\lambda_2(V(k)) \geq \lambda_2(V(k')) \Rightarrow G(k) \geq G(k')$ for all $k, k' \in T_n$.*

The above definition of voronoi bias allows sampling distributions that are more general *i.e.* distributions that satisfy $\psi(A) \geq \psi(B)$ for $\lambda_2(A) \geq \lambda_2(B)$ for any bounded closed region $A, B \in \mathbb{R}^2$. It is difficult in general to obtain a closed form solution for $G(x)$.

B. Analysis of ℓRRT

Since the ℓRRT has edges of length exactly r, the tree can only be (δ, ε)-space filling with $\varepsilon = r$, whereas in a RRT ε can be less than r as we add more nodes to the tree. For the ℓRRT of step size $r = 2$ embedded in a square of length 100, the constants α and β were leanred to be 0.9981 and 0.00009 respectively, which are almost equal to

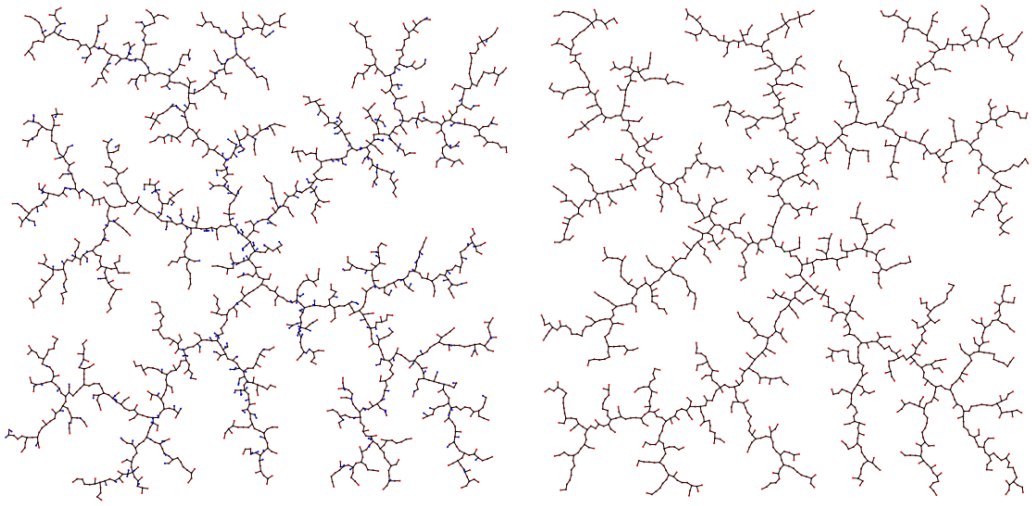


Fig. 9. Embedding of RRT(left) and ℓRRT(right) of edge length $r = 2$ in a square of length 100. The number of nodes in the lrrt is 1687 (2000 sampled points) while the number of nodes in the RRT is 2000.

the parameters leanred for the corresponding RRT. Fig. 8 plots δ for the two trees and it is clear that both trees take almost the same number of nodes to completely fill a space although the number of points sampled in case of the ℓRRT is more. Fig. 9 compares an embedding of a RRT and an ℓRRT.

For the ℓRRT the parameter p in the definition of voronoi bias is given by $p = 1 - \lambda_2(C_n)/\lambda_2(A)$. Although we observed that ℓRRTs have voronoi bias, it remains to be proved if the bias is strong or weak. In subsequent sections we model the multi-robot system as a ℓRRT.

CHAPTER V

IMPLEMENTATION

A. Implementation requirements

Building a structure that grows towards unexplored regions inherently calls for a mechanism to sample the environment for unexplored regions. Sampling can be explicit or implicit. For explicit sampling a global knowledge of the environment is needed which can be accomplished by using more advanced sensing equipment like overhead cameras. But such solutions are unattractive and impractical due to lack of scalability and in most cases require offline computation. On the other hand implicit sampling is more robust and scalable but comes at the cost of reduced accuracy. In our approach we implicitly sample the environment by having robots wander in the environment: large open free spaces have greater likelihood of having wandering robots spiralling out and joining the tree. Other approaches can be employed for implicit sampling in which agents have the ability to encode information into the environment *e.g.* by using a pheromone like mechanism to bias exploration towards regions of low pheromone density (large unexplored spaces will have less pheromone density than regions already explored).

Another requirement for implementing the approach presented in this paper is the ability of robots or agents to find the nearest neighbor and extend the tree towards the randomly sampled point. The spiralling maneuver fulfils the above two requirement at the same time. Thus there is no one-to-one mapping between the steps of the actual RRT algorithm and the operations performed by robots. An alternative, albeit far less accurate, approach to the spiralling maneuver would be a simple random walk maneuver. Although a random walk will not necessarily find the correct nearest

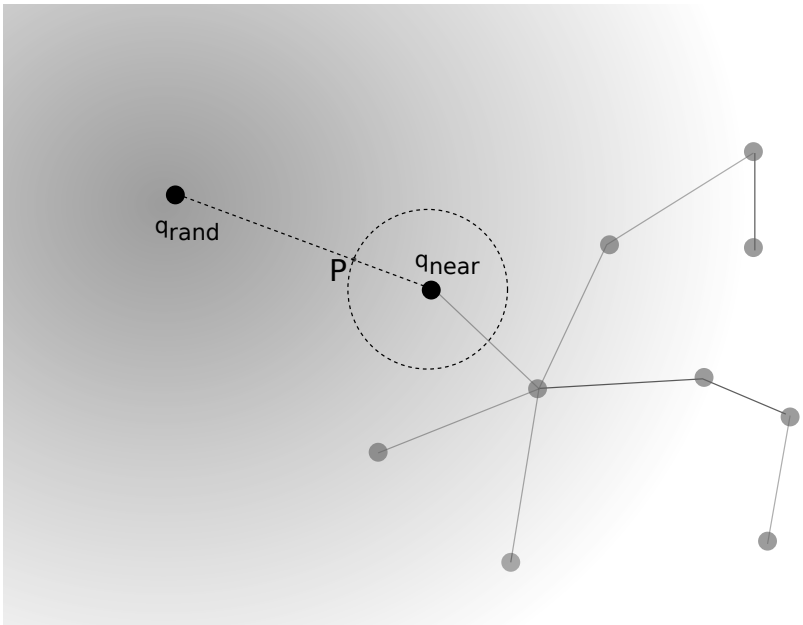


Fig. 10. The circle around the point q_{near} represents positions from which the robot executing a random walk maneuver can join q_{near} in one step. The probability of the robot joining from point P is higher than all other points in the circle because the expected hitting time for the point P is minimum.

neighbor with probability one, it will find the correct nearest neighbor in expectation because of the fact that the expected hitting time of the nearest neighbor is minimum over all other nodes. The random walk maneuver can also be used to extend the tree towards the randomly sampled point, again in expectation as illustrated in Fig. 10.

B. Actual 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

Fig. 11. A typical trial using 7 iRobot Create robots with $\Delta q = 2$. 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.

an 3.66m $\times$ 5.05m 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. 11 is an example from such a trial.) And this combination proved to be the most efficient in terms of the time taken for all robots to join the tree.

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 with Δq as 1, but alignment for the ExtendEdge operation proved challenging when $\Delta q > 1, n > 2$ as communicative bumps slowly pushed robots eout of the tree.

CHAPTER VI

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 ultimate 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, $\psi(x) : x \to [0, 1]$ for all $x \subseteq A$, that determines the probability, $\mathbf{Pr}\left[q_{\text{rand}} \in x\right]$, 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 $\mathbf{Pr}\left[q_{\text{rand}} \in x\right] = \lambda_2\left(x\right)/\lambda_2\left(A\right)$, where $\lambda_2\left(.\right)$ is the area of a region.

A complication arising from of the distributed construction of the data-structure is that multiple operations can proceed in parallel, and they do not map to a single atomic step in the original algorithm. The steps involving 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 § C, 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 $r = 2(k + 1)\rho$.

We simplify the analysis by assuming that the align and trace operations are instantaneous and thus the align and trace states in Fig. 4 can be merged with the wandering state. The simplified controller with high level behaviors is in Fig. 12. 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

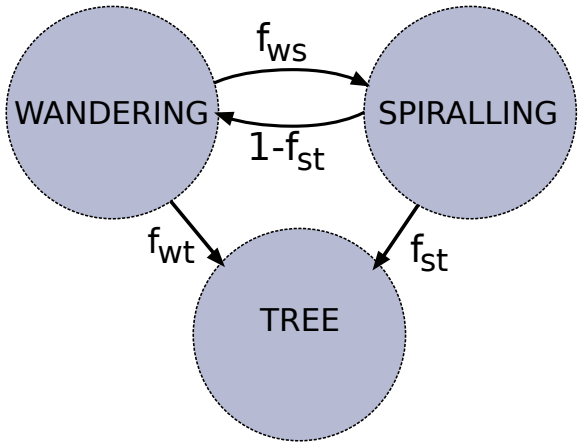


Fig. 12. Simplified controller states of a robot along with transition rates.

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 explore the dynamics of the tree to compute f_{st}, and then finally develop the model characterizing 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: $\mathbf{Pr}\left[X_{t+\tau} = y | X_t = x\right] = 1/\lambda_2\left(A\right)$ for all $x, y \in A$. Thus, if the wandering robot is uniformly distributed at time t *i.e.,* $\mathbf{Pr}\left[X_t = x\right] = 1/\lambda_2\left(A\right)$, 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 8 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, $n = \left\lceil \sqrt{\lambda_2\left(A\right)/(\vartheta \Delta t)^2} \right\rceil$. 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 [20].

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

$$\tau_{mix}(\varepsilon) \leq d^2 n^2 \log_2(\varepsilon^{-1}), \tag{6.1}$$

where ε is the variation distance from the stationary distribution.

Thus, the parameter τ is given by:

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

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 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}:

$$f_{ws} = \frac{1}{N\tau}, \tag{6.2}$$

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 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: $N_s(t), N_w(t)$ and $N_r(t)$ where representing the number of spiralling, wandering, and tree robots, at time t. With N robots, the number of nodes in the tree $N_r(t) = N-(N_s(t)+N_w(t))$. Since the robots are reactive and memoryless, the actions of each robot can be described by a Markov process with the tree state being an absorbing state. The transition rate of a robot

transitioning from wandering to the spiralling is given by f_{ws} as mentioned earlier and is obtained from (6.2). 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

$$r(t) = \sqrt{\frac{a\rho}{\pi}}t. \tag{6.3}$$

Note that (6.3) depends on how the robots trace out their spirals. Our controller maintains constant angular velocity, by increases the tangential velocity linearly.

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,

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

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*

$$\mathbf{E}[D_n(x)] = \frac{1}{2\sqrt{n}} \int_Q f(x)^{-1/2} dx \tag{6.4}$$

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 probability is given by:

$$\mathbf{Pr}[D_n(x) > k] = \int_Q (1 - U(x,k))^n dx \text{ where } U(x,k) = \int_{D(x,k)} f(x) dx$$

Now $U(x,k) \approx \pi f(x) k^2$, and $(1-U)^n \approx e^{-nU}$. So,

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

□

Since, there are $N - N_r(t)$ robots distributed uniformly over the region $A/B_{N_r(t)}$, the mean free path of a robot follows from (6.4) and is then given by

$$
\begin{aligned}
d_{free}(N_r) &= \frac{1}{2\sqrt{N - N_r(t)}} \int_{A/B_{N_r(t)}} \frac{1}{\sqrt{\lambda_1(A) - \lambda_1\left(B_{N_r(t)}\right)}} dx \\
&= \left(\frac{1}{2\sqrt{N - N_r(t)}}\right) \left(\frac{\lambda_1\left(A/B_{N_r(t)}\right)}{\sqrt{\lambda_1(A) - \lambda_1\left(B_{N_r(t)}\right)}}\right) \\
&= \frac{1}{2}\sqrt{\frac{\lambda_1(A) - \lambda_1\left(B_{N_r(t)}\right)}{N - N_r(t)}}. \qquad (6.5)
\end{aligned}
$$

So, if the average relative speed of a robot is v_{rel}, then the mean free time is

$$
\tau_{free}(N_r) = \frac{1}{2v_{rel}}\sqrt{\frac{\lambda_2(A) - \lambda_2\left(B_{N_r(t)}\right)}{N - N_r(t)}}. \qquad (6.6)
$$

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 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:

$$
\mathbf{Pr}\left[S \leq t\right] = \begin{cases} 1 - e^{-\lambda_t t}, & t \geq 0 \\ 0, & t < 0. \end{cases} \qquad (6.7)
$$

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.,* as defined in Eq. 4.9. 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:

$$\begin{aligned} J(x) &= p_v(t)\mathbf{Pr}\left[Q_{t+r^{-1}(x)} = \text{tree} | Q_t = \text{spiralling}\right] \\ &= p_v(t)\mathbf{Pr}\left[S > r^{-1}(x)\right] \\ &= p_v(t)e^{-\lambda_t\sqrt{\frac{\pi}{a\rho}}x}. \end{aligned} \tag{6.8}$$

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:

$$\widehat{J}(x) = 1 - e^{-\lambda_t\sqrt{\frac{\pi}{a\rho}}x}, \qquad j(x) = \frac{d}{dx}(\widehat{J}(x)) = \lambda_t\sqrt{\frac{\pi}{a\rho}}e^{-\lambda_t\sqrt{\frac{\pi}{a\rho}}}. \tag{6.9}$$

Thus, the instantaneous transition rate of transitioning from the spiralling state to the tree state, f_{st}, is given by (using $\gamma = \sqrt{(a\rho)/\pi}$):

$$
\begin{aligned}
f_{st} &= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \mathbf{Pr}\left[Q_{t+\Delta t} = \text{tree} | Q_t = \text{spiralling}\right] \\
&= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \left[\int_0^{\gamma \Delta t} \int_0^{2\pi} j(x) dx d\theta \right] \\
&= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \left[\int_0^{\gamma \Delta t} \int_0^{2\pi} \frac{\lambda_t}{\gamma} e^{-\frac{\lambda_t}{\gamma} x} dx d\theta \right] \\
&= 2\pi \lim_{\Delta t \to 0} \left[\frac{1 - e^{-\lambda_t \Delta t}}{\Delta t} \right] \\
&= 2\pi \lambda_t \\
&= 4\pi v_{rel} \sqrt{\frac{N - N_r(t)}{\lambda_2 \left(A / B_{N_r}\right)}}
\end{aligned} \tag{6.10}
$$

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 $t + \Delta t$. Note how the last line shows that the transition rate depends on collision parameter λ_t.

Let p_w (p_s, p_r) be the probability that a robot is in wandering (spiralling, tree) state at time t. Then, the following stochastic master equation gives the change in probabilities with time:

$$
\begin{aligned}
\frac{d\left(p_w\right)}{dt} &= p_s(t)(1 - f_{st}) - p_w f_{ws}, & (6.11) \\
\frac{d\left(p_s\right)}{dt} &= p_w f_{ws} - p_s(t) f_{st} - p_s(t)(1 - f_{st}), & (6.12) \\
\frac{d\left(p_r\right)}{dt} &= p_s(t) f_{st}. & (6.13)
\end{aligned}
$$

If N is the total number of robots, then $N \cdot p_w$ gives the fraction of robots that are in the wandering state. Averaging (6.11)–(6.13) over the number of robots, N, yields the following rate equations describing the evolution of the average number of

wandering robots, spiralling robots and tree robots respectively with time.

$$\frac{d\left(\widetilde{N}_w(t)\right)}{dt} = \widetilde{N}_s(t)(1-f_{st}) - \widetilde{N}_w(t)f_{ws}, \tag{6.14}$$

$$\frac{d\left(\widetilde{N}_s(t)\right)}{dt} = \widetilde{N}_w(t)f_{ws} - \widetilde{N}_s(t)f_{st} - \widetilde{N}_s(t)(1-f_{st}), \tag{6.15}$$

$$\frac{d\left(\widetilde{N}_r(t)\right)}{dt} = \widetilde{N}_s(t)f_{st}. \tag{6.16}$$

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:

$$f_{wt} \leq \frac{(k+1)}{\tau_{free}(N_r)} = \left(\frac{k+1}{N_r}\right)\frac{2v_{rel}\sqrt{N-N_r}}{\lambda_2\left(A/B_{N_r}\right)}. \tag{6.17}$$

Since a spiralling robot can initiate and 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:

$$f_{st}{}' = \frac{f_{st}}{k+1} \tag{6.18}$$

and the rate equations for the general case of $k \geq 0$ is given by:

$$\begin{aligned}
\frac{d\left(\widetilde{N}_w(t)\right)}{dt} &= \widetilde{N}_s(t)(1-f'_{st}) - \widetilde{N}_w(t).f_{ws} - \widetilde{N}_w(t)f_{wt}, && (6.19)\\
\frac{d\left(\widetilde{N}_s(t)\right)}{dt} &= \widetilde{N}_w(t)f_{ws} - \widetilde{N}_s(t)f_{st}{}' - \widetilde{N}_s(t)(1-f_{st}{}'), && (6.20)\\
\frac{d\left(\widetilde{N}_r(t)\right)}{dt} &= \widetilde{N}_s(t)f'_{st} + \widetilde{N}_w(t)f_{wt}. && (6.21)
\end{aligned}$$

When $k > 0$, expression (6.16) gives a lower estimate of the average number of tree robots at time t, while (6.21) is an overestimate of the average number of tree robots. Next, we explore the Voronoi bias of the physical tree formed by robots based on the definitions introduced in chapter IV.

D. Voronoi bias of physical tree

In § A 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 density function, $j(x)$, on the bias, broadening the notion of a Voronoi bias.

Let $J(V_k) = \int_{V_k} j(x)dx$, where V_k represents the voronoi cell of the node $k \in T_n$. Then the effective sampling distribution, ψ', is given as $\psi'(V_k) = \psi(V_k)J(V_k)$. As shown in § 2, for there to be a strong bias $\psi'(x)$ has to be uniform. 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. In general, the joining distribution $J(V_k)$ depends not 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 how "well-behaved" the distribution of Voronoi cells is. It is clear that the weaker definition of Voronoi bias 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 $O(\sqrt{N})$.

E. Accuracy of AddEdge operation

In chapter III we introduced the AddEdge operation and claimed that an edge is added in the direction of the randomly sampled point q_{rand}. Since edges are initiated by spiralling robots, which are added in place; the accuracy with which the tree is grown towards the randomly sampled point is determined by the angle of approach of the spiralling robot with respect to its nearest neighbor. The maximum angle of approach, η, of a spiralling robot is illustrated in Fig. 13 and is given as follows:

$$\eta = \cos^{-1}\left(\frac{(8\rho^2 + 4\rho x) - (b^2 + 2xb)}{8\rho^2 + 4\rho x}\right) \tag{6.22}$$

In our implementation the separation between successive rings is the diameter of the robot, 2ρ. Thus the maximum angle of approach is given by:

$$\eta = \cos^{-1}\left(\frac{\rho}{2\rho + x}\right). \tag{6.23}$$

From Eq. 6.23, the maximum angle of approach is 90°. Thus an edge is extended between 0° and 90° of the randomly sampled point. But, if the spiralling robots randomly trace out clockwise and anti-clockwise spirals with equal probability then

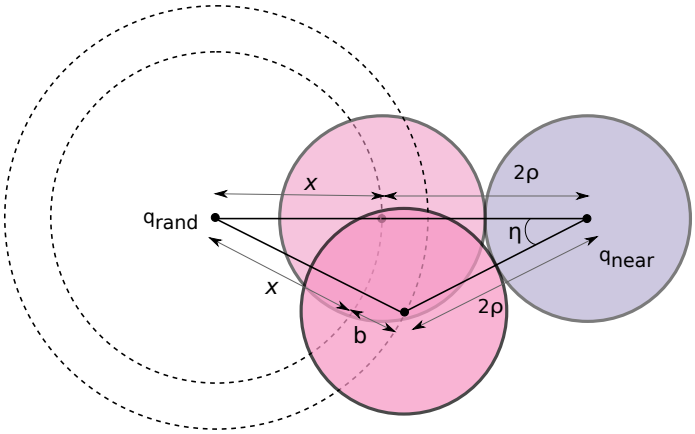


Fig. 13. Maximum approach angle, η, of a spiralling robot tracing a spiral with separation b between successive spirals and spiralling out from a distance $x + 2\rho$ from the tree.

on an average the edge is extended in the direction of the randomly sampled point.

CHAPTER VII

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. 14 shows results averaged over 10 trials for each value of k , along with snap-shots from the simulation in Fig. 15. For $k = 1$ the initial size of the tree was 11, for $k = 2$ the initial size of the tree was 16 and for $k = 3$ the size was 21. The initial size were choosen such that all trees had 6 nodes (vertex robots) including the initial node.

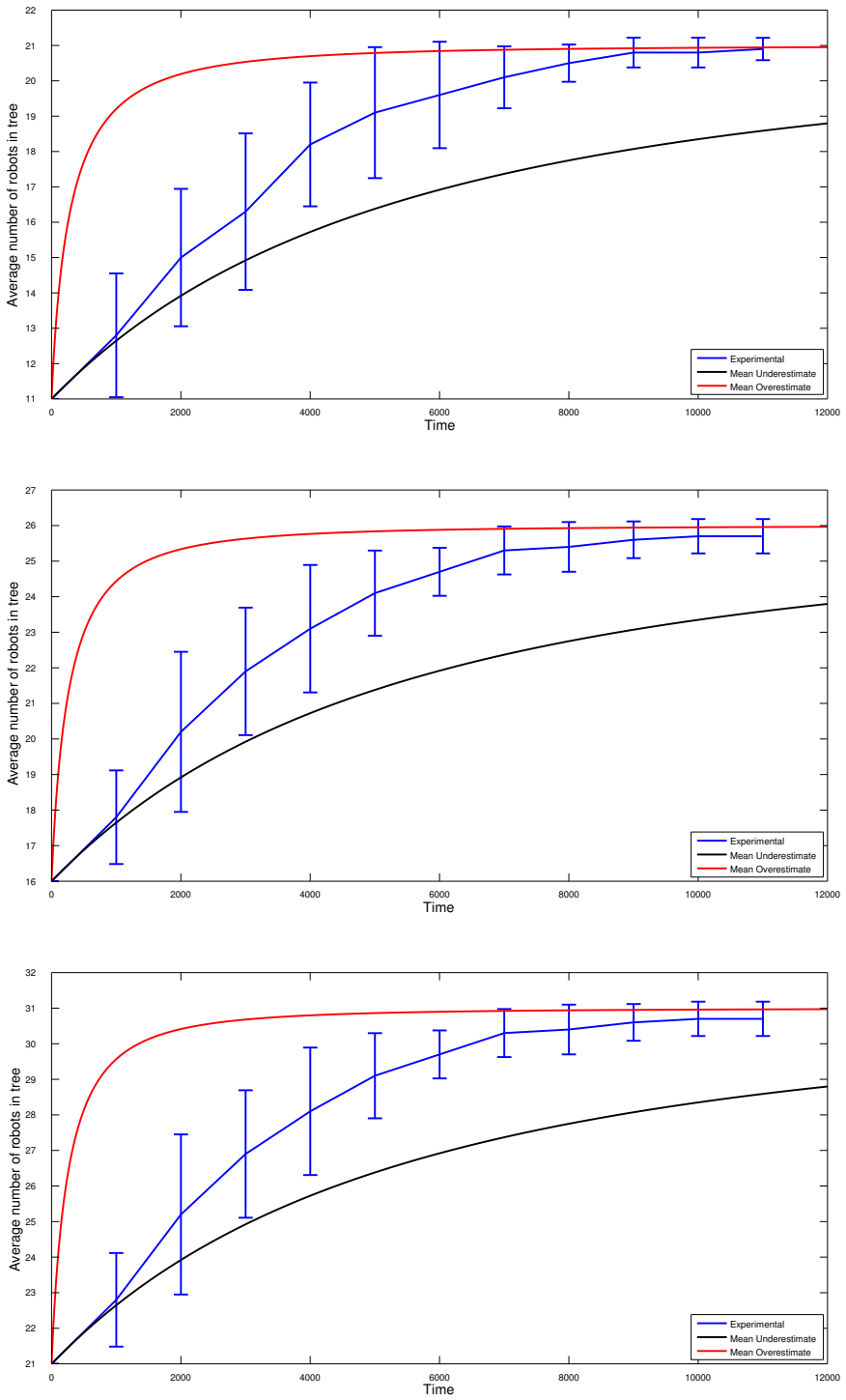


Fig. 14. Plot showing the underestimate (black) and overestimate (red) of the average number of tree robots at a given time for each of the case of $k = 1$ (top), $k = 2$ (middle), $k = 3$ (bottom). The average number of robots in the tree calculated experimentally are shown in blue.

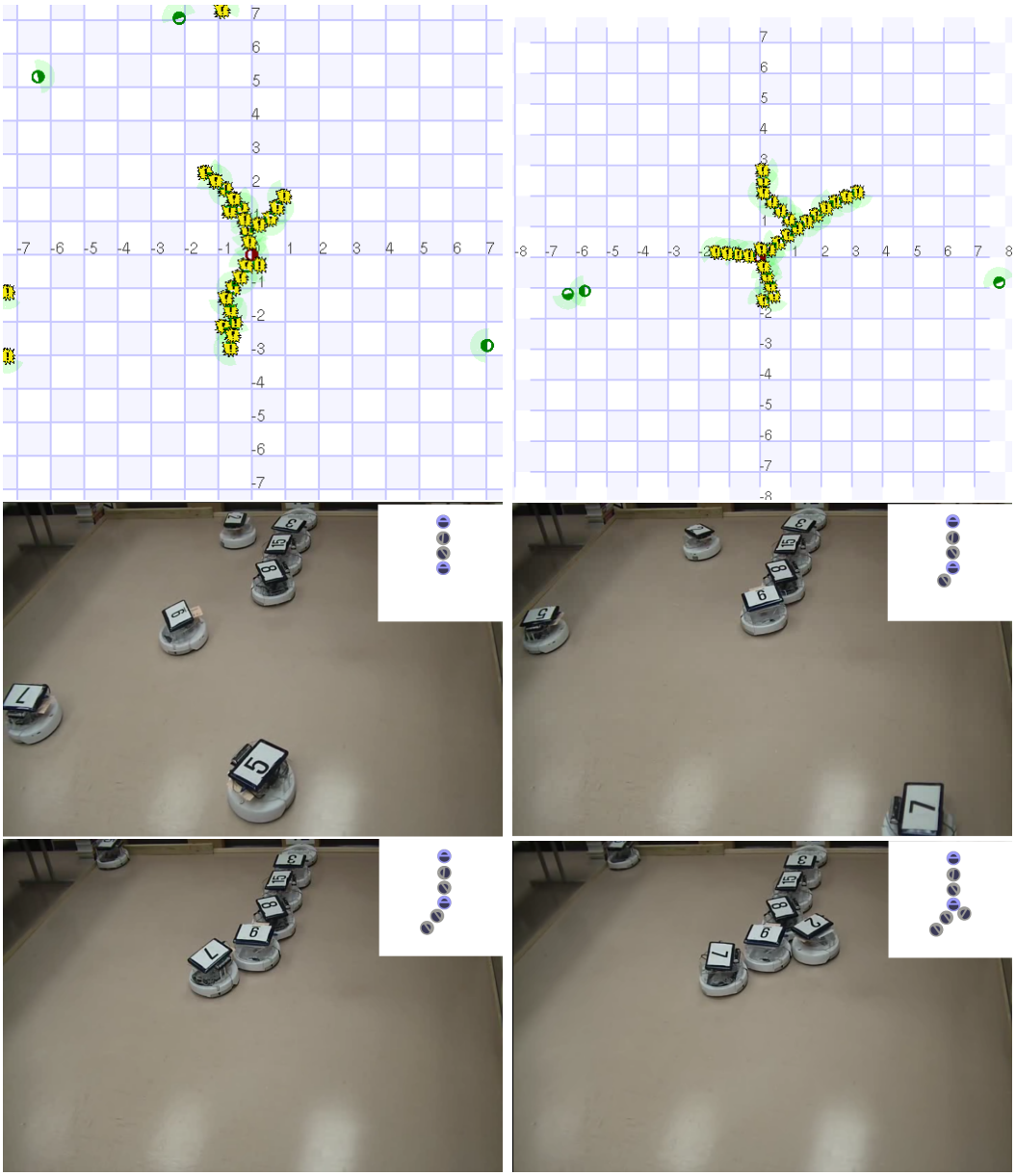


Fig. 15. Two snapshots from simulation experiments (top) showing trees of edge length 0 and 3 respectively and snapshots from a physical robot experiment (bottom) of edge length 2 with the logical tree inset.

CHAPTER VIII

DISCUSSION

In the previous section we presented results from simulation experiments and compared it with the theoretical overestimate and underestimate of the mean, finding good agreement between the theoretical model and experimental data. In our analysis we made a few simplifying assumptions regarding the environment, namely: the region is convex and obstacle free and that spiralling robots do not collide with the boundary. While the latter assumption can be easily addressed, the former is more difficult owing to the fact that non-convex regions and obstacles introduce *shadows* and *holes* in the region which have complex influences on the joining distribution $j(x)$. Boundary effects can be taken into account by considering the following simplifying assumption: since at any time the free space available is given by A/B_{N_r}, half of the region can be assumed to be closer to the boundary and the other half closer to the tree. Thus, the transition probability f_{st} can be multiplied by a factor of 0.5 and the results from using the modified transition probability is shown in Fig. 16.

Although in § B we presented a method that guarantees uniform sampling, the value of p_{ws} is chosen out of more practical considerations to improve performance. The presented model is general enough to be applied to more capable robots.

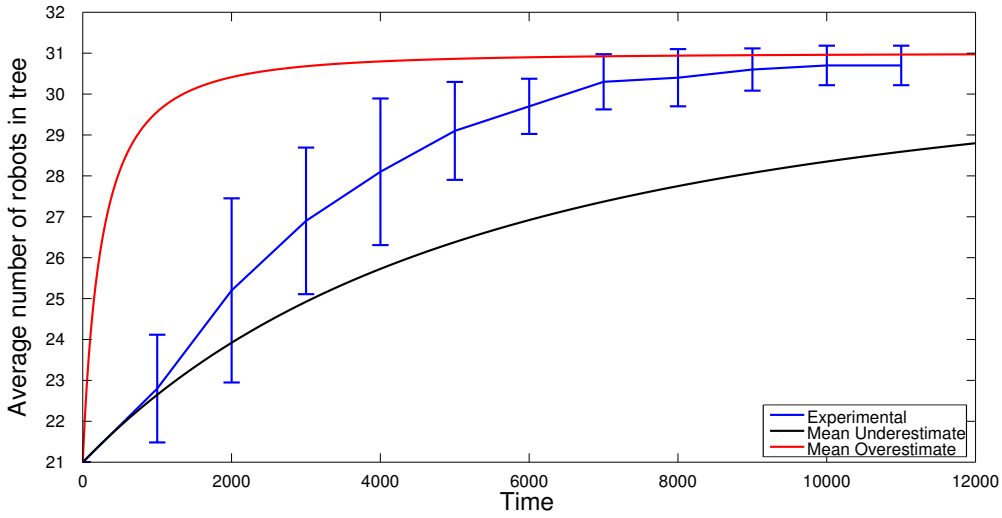


Fig. 16. Plot showing experimental data (blue), the theoretical underestimate and overestimate for k = 2 after considering boundary effects.

CHAPTER IX

CONCLUSION AND FUTURE WORK

Although we demonstrated that ℓRRTs and RRTs are equally good when it comes to filling a space the voronoi bias of ℓRRTs remain to be quantified. It also remains to be seen if our approach can be generalized to 3 dimensions satisfying the requirements charted out in the implementation section; although preliminary investigation suggests that it is difficult to come up with a maneuver for finding the nearest neighbor in 3-D. Another direction for extending the current work is to explore the use of heterogeneous robots: having specialized robots for completing incomplete edges.

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 to be carried over to the state of the robots themselves. In this particular case, we have extended, broadened, or 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. on Simulation of Adaptive Behavior*, 1996, pp. 625–634.

[2] S. Nouyan, A. Campo, and M. Dorigo, "Path formation in a robot swarm," *Swarm Intelligence J.*, vol. 2, no. 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 J.*, vol. 5, pp. 73–96, 2011.

[4] S. M. Lavalle and J. J. Kuffner Jr., "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000, pp. 293–308.

[5] B. R. Donald, J. Jennings, and D. Rus, "Minimalism + distribution = supermodularity," *J. of Experimental 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," *Autonomous Robots*, vol. 11, no. 3, pp. 319–324, Nov. 2001.

[7] F. Lamiraux and J. P. Laumond, "On the expected complexity of random path planning," in *Proc. IEEE Int. Conf. Robot. & Autom.*, 1996, pp. 3306–3311.

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

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

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

[11] K. Sekiyama and T. Fukuda, "Modeling and controlling of group behavior based on self-organizing principle," in *Proc. IEEE Int. Conf. Robot. & Autom.*, Apr. 1996, pp. 1407–1412.

[12] K. Sugawara and M. Sano, "Cooperative acceleration of task performance: Foraging behavior of interacting multi-robots system," *Physica D: Nonlinear Phenomena*, vol. 100, no. 3/4, pp. 343–354, Feb. 1997.

[13] K. Lerman and A. Galstyan, "A general methodology for mathematical analysis of multi-agent systems," Tech. Rep. ISI-TR-529, USC Information Sciences Institute, Marina del Rey, CA, 2001.

[14] A. Martinoli, K. I. Easton, and W. Agassounon, "Modeling of swarm robotic systems: A case study in collaborative distributed manipulation," *Int. J. of Robotics Research*, vol. 23, no. 4, pp. 415–436, 2004.

[15] T. W. Mather and M. A. Hsieh, "Analysis of stochastic deployment policies with time delays for robot ensembles," *Int. J. of Robotics Research*, 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 *Proc. IEEE Int. Conf. Robot. & Autom.*, May 2011, pp. 378–385.

[17] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep. TR 98-11, Computer Science Dept., Iowa State University, Ames, IA, 1998.

[18] J. J. Kuffner Jr. and S. M. Lavalle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. & Autom.*, 2000, pp. 995–1001.

[19] "CGAL, Computational Geometry Algorithms Library," 2012, http://www.cgal.org.

[20] D. A. Levin, Y. Peres, and W. L. Elizabeth, *Markov chains and mixing times*, American Mathematical Society, Providence, RI, 2006.

VITA

Name Asish Ghoshal

Address Department of Computer Science,
301 Harvey R. Bright Building,
College Station, TX 77843-3112

Email Address aghoshal@cs.tamu.edu

Education

- Master of Science, Computer Science, May 2012, Texas A&M University, College Station, TX
- Bachelor of Technology, Computer Science and Engineering, May 2008, National Institute of Science and Technology, Brahmapur, Orissa, India