Symmetric Rendezvous Search on the Line with an Unknown Initial Distance

Deniz Ozsoyeller, Andrew Beveridge and Volkan Isler

***Abstract*—In the rendezvous search problem, two robots at unknown locations must successfully meet somewhere in the environment. We study the symmetric version of the problem in which they must use the same strategy. We provide a new algorithm for the symmetric rendezvous problem on the line. Our symmetric strategy has a competitive ratio of 17.686 for total distance traveled and a competitive ratio of 24.843 for total time. Both are improvements over the previously best known algorithm, which has (time and distance) competitive ratio of 26.650. Our algorithm can be adapted for bounded linear environments and simple closed curves with the same performance guarantees. It is also robust with respect to errors in motion and differences in robots' starting times. We confirm our theoretical results through simulations, and show that our algorithms are practical by reporting the results of real robot deployments in indoor environments.**

***Index Terms*—rendezvous search, symmetric rendezvous, linear search problem, lost cow problem**

I. Introduction

The goal of rendezvous search is to design strategies for robots whose locations are unknown to each other to meet as quickly as possible. In this paper, we study the rendezvous search problem with two robots in a one-dimensional environment. For example, robots dropped off from a plane may be trying to find each other after deployment in linear environment, such as a road, a corridor or a river. This environment could be infinite, circular or bounded. However, no obstacles that disconnect the environment can reside between the robots, otherwise rendezvous will never occur. The rendezvous search problem has attracted significant attention from the research community and studied in various environments such as line, graph, circle (ring) and plane. A comprehensive introduction can be found in the survey [1], and the monograph [2].

There are two primary versions of the rendezvous search problem. In *asymmetric rendezvous search*, the players can choose separate roles in advance and execute distinct strategies. For example, one can remain stationary while the other actively searches. In the second version, called *symmetric rendezvous search*, the players must execute the same strategy. In the literature, both of these versions have been studied on various environments such as line, circle, graph and plane. The symmetric version is appealing for robotics applications because it eliminates the need for implementing a different version of the strategy on each robot.

Deniz Ozsoyeller is with the Department of Computer Engineering, Izmir University, Turkey and the International Computer Institute, Ege University, Turkey. deniz.ozsoyeller@izmir.edu.tr

Andrew Beveridge is with the Department of Mathematics, Statistics and Computer Science, Macalester College. abeverid@macalester.edu

Volkan Isler is with the Department of Computer Science and Engineering, University of Minnesota. isler@cs.umn.edu

Suppose that two robots start at locations x and y in an environment Q. Let $d(x, y)$ be the distance between these points. For a rendezvous strategy $\mathcal{S}$, let $S_i(x, y)$ denote the (expected) distance traveled by robot i before rendezvous, $i = 1, 2$. The efficiency of a rendezvous strategy $\mathcal{S}$ is often measured by its competitive ratio

$$\max_{x,y \in Q} \frac{S_1(x, y) + S_2(x, y)}{d(x, y)} \tag{1}$$

Here the denominator is the minimum possible distance traveled before rendezvous. The competitive ratio of $\mathcal{S}$ is the worst case deviation of the performance of $\mathcal{S}$ from this optimal behavior. A strategy is said to be *competitive* if its competitive ratio is a constant. We note that fixing a rendezvous location in advance is not an efficient strategy: the robot deployment locations could be quite close to each other other while being very far from the predetermined rendezvous location. In equation (1), the competitive ratio is given with respect to the total distance traveled. Other measures such as the maximum distance traveled can also be used.

When the robot strategies do not involve idle times, the distance traveled is the same as amount of the time elapsed. Herein we consider randomized strategies that intersperse robot movements with robot idle times. Therefore, we consider two competitive ratios: the distance competitive ratio and the time competitive ratio. Since robot motion is more expensive than robot idle time, we will focus on minimizing the distance competitive ratio of our fixed strategy. For symmetric strategies, the distance competitive ratio also equals

$$\max_{x,y \in Q} \frac{2S_1(x, y)}{d(x, y)} = \max_{x,y \in Q} \frac{S_1(x, y)}{d(x, y)/2}.$$

We use this formulation in our analysis below.

Rendezvous on the line has been well studied for both symmetric [1], [3]–[7] and asymmetric [8]–[12] versions. In addition to its theoretical importance, rendezvous on the line is important for robotics applications: the line can be used to model environments such as railroads, cables or long corridors. In the original symmetric formulation, the players start knowing the initial distance $2d$. For this formulation, Alpern [1] introduced the following strategy: at the beginning of each period, each player independently chooses a random direction to call forward (F) and goes d units forward and then $2d$ units in the opposite backwards (B) direction. This randomized 1F2B moving pattern repeats until rendezvous is achieved. Over the years, the analysis of this algorithm has been improved, so that guaranteed competitive ratio has been decreased from Alpern's initial value of 5 to 4.5678 by Anderson and Essegaier [3], to 4.4182 by Baston and Gal [9], and to 4.39306 by Uthaisombut [5].

In [9], Baston and Gal considered a symmetric rendezvous formulation in which the initial distance is drawn from some distribution with expected value $E[2d] = \mu$. They provide a symmetric algorithm with expected meeting time 13.325μ, corresponding to competitive ratio 26.650. For a distance distribution with maximum distance D, Baston [4] developed an algorithm with competitive ratio $1.71 + \mu/2D$.

Herein, we consider rendezvous problems on the line with no information about the initial distance. The players do not share a global frame. The unknown initial distance case has been investigated for asymmetric rendezvous [7], [12], but it has not been well studied for the symmetric case. Baston and Gal's algorithm from [9] is agnostic of the distance distribution and is therefore a 26.650-competitive algorithm (for both distance and time) for this formulation.

The contributions of this paper are as follows. We provide a new algorithm for linear symmetric rendezvous at an unknown (and arbitrary) distance. This algorithm is an improvement over the Baston and Gal algorithm: it is 17.686-distance competitive and 24.843-time competitive. We provide simulation results and experimental results from a robot deployment. An interesting aspect of our algorithms is the randomization of the distance traveled (in addition to the motion pattern) at each round. Our simulations show that this randomization makes the algorithm robust with respect to uncertainty in motion e.g. due to errors in odometry.

Our algorithm (and its theoretical analysis) assumes that the robots start at the same time. We investigate the significance of this assumption in simulations which suggest that the competitive ratio does not increase significantly even when the robots start at different times. Finally, we adapt the algorithm for two real world rendezvous tasks: when the linear environment is of bounded length, and when the linear environment is a simple closed curve (such as a circular loop).

The paper is organized as follows. In Section II, we present an overview of related work. Next, we review a related online problem in Section III-A, then study the linear symmetric rendezvous problem in Sections III-B and III-C. We present the simulations results where the robots start executing the algorithm at the same time in Section IV-A and at different times in Section IV-B. In Section V, we present extension to circular and bounded environment and report results from real experiments in Section VI.

II. Rendezvous versus Rendezvous Search

In the robotics literature, there are two classes of rendezvous problems. The first version concerns robot tracking and navigation towards a moving object (target) where the agents can observe each other's state. Therefore, the emphasis is on control-theoretic aspects such as combining the kinematics equations of the robot and the target. The target can be another mobile robot, a satellite, a moving convoy or a human. Sometimes, the desire to rendezvous is one-sided (for example, an agent who wants to meet up with a moving convoy). Moreover, the robot and the target can have different features such as size, velocity, orientation, etc. One example of this type of rendezvous is the optimal control problem for minimum control energy needed to reach a moving target [13].

The *parallel navigation law* is one of the most widely used closed loop controls for the rendezvous of the robot with the target. This approach combines the kinematics equations of the robot and the target with geometric rules. The robot moves in lines that are parallel to the initial line of sight (LOS), which is the relative position vector connecting the robot to the target. The parallel navigation rule states that the angle of the LOS of the robot to the target remains constant at all times [14]. Therefore, the interception is considered only in the direction of LOS. The robot may or may not know the environment and the target maneuvers in advance. Usually, the robot has a sensory system to detect obstacles and obtain the necessary information on the target such as its position, linear velocity and orientation. The information gained on the target (and the obstacles) is used in the rendezvous guidance algorithm.

In this paper, we focus on the rendezvous *search* problem. Similar to the rendezvous tracking and navigation problem above, this deployment can be thought as establishing LOS between robot-1 to robot-2 in a given environment. However, the remaining formulation is quite different. The heart of the problem is the lack of state information. The robots cannot communicate, and do not have a (long range) sensory system that allows them to determine the position of the other robot (though they can detect rendezvous). The robots do not necessarily know their current location and they do not (and cannot) know the initial distance or direction to the other robot.

Another difference is that *both* robots actively work to achieve rendezvous, compared to having an active robot and an agnostic target. A third difference is that the robots have identical properties; for example, they have the same size and velocity. Our final, key limitation is that in symmetric rendezvous search, the robots must execute the same strategy.

III. Symmetric Rendezvous On The Line

The rendezvous problem generalizes the *linear search problem*, also known as the *lost cow problem*. In this formulation, a single robot tries to find an unknown stationary object. This linear search problem was originally solved in [15] and that solution was rediscovered in [16]. This result was placed into the framework of online algorithms in [17] (where it was also generalized to M infinite rays radiating from a common starting point). Our symmetric rendezvous algorithm generalizes and draws inspiration from the lost cow problem. Therefore, we review that linear search problem and then describe of our algorithm and analyze its performance.

A. The Lost Cow Problem

Suppose that a near-sighted cow tries to find the only gate in a long, straight fence. This gate is located at an unknown initial distance d, either to the left or the right of the cow. Any deterministic online algorithm for locating the gate is equivalent to alternately searching to the left and then to the right of the cow's initial location $x = 0$. Such a spiraling algorithm can be defined by a sequence of non-negative numbers $f_1, f_2, \ldots$, where f_i is the distance explored to left (right) of the cow's initial position in odd (even) round

i. For convenience, we set $f_0 = 0$. In the ith round, the cow travels a total distance of $f_{i-1} + f_i$.

The deterministic *Spiral LostCow algorithm* uses a doubling strategy $f_i = 2^{i+1}$. Spiral LostCow is a 9-competitive algorithm, and this is the best possible performance for a deterministic online algorithm. This competitive ratio can be improved by introducing some randomness. The *SmartCow algorithm* follows the same zig-zag strategy, except the distance traveled in round i is $r^{i+\epsilon}$, where $\epsilon \in [0, 1)$ is chosen uniformly at the start of the algorithm. This dash of randomness smooths out the worst case analysis. Beck and Newman [15] proved that the value $r \approx 3.591$ guarantees competitive ratio of 4.591, which is best possible. This result was rediscovered by Kao et al. [18].

We now compare the rendezvous problem with the lost cow problem. Instead of one mobile cow and a stationary gate, we now have two mobile cows, initially separated by a distance of $2d$. The goal of the cows is to meet up anywhere along the fence. The symmetric version of this problem requires that both cows perform the same search algorithm. This forbids deterministic strategies, since their synchronized movement would prevent the cows from ever meeting.

B. The Symmetric Rendezvous Algorithm

We present our algorithm for symmetric rendezvous for an unknown initial distance. Our algorithm uses randomization to break the symmetry between the robots. We also use a motion pattern which combines rounds $2i$ and $2i + 1$ of the Spiral LostCow algorithm into a single round. This will guarantee that, once the robot's displacement is at least d units, the probability of meeting in every subsequent round is $1/2$.

We now describe the randomized itinerary of the robots. Let $2d$ be the initial distance between two robots, as shown in Figure 1. Without loss of generality, robot-1 is located at $x = 0$ and robot-2 is located at $2d$ (though they are unaware of their relative positions). We represent the configuration of the robots as a point in $\mathbb{R}^2$, so that the initial configuration is $(0, 2d)$. The robots do not know their initial distance or the direction leading to the other robot. For omniscient robots, the best offline algorithm would be for them to move towards each other and meet at configuration (d, d). Hence, our competitive ratio will be calculated in comparison with distance d.

To each robot, we associate a non-negative sequence $f_{-1}, f_0, f_1, f_2, \ldots$, where $f_{-1} = 0$ (for convenience) and

$$f_i = r^{i+\epsilon} \text{ for } i \geq 0$$

where $r > 0$ is the expansion radius (to be optimized later) and $\epsilon \in (0, 1]$ is a uniformly distributed random variable. The robots use the same expansion radius r, but they choose their values for ϵ independently at the start of the algorithm. The robot uses this random ϵ value throughout the algorithm (choosing a new ϵ value in each round does not improve the performance). As in the SmartCow algorithm, this random variable mitigates the worst case competitive ratio, giving an improved performance guarantee. Indeed, in [19], we presented a variant of the $\mathcal{SR}$ algorithm without the random ϵ. That algorithm was shown to be 19.166 distance competitive and 26.888 time competitive. The addition of the randomized parameter ϵ results in better performance.

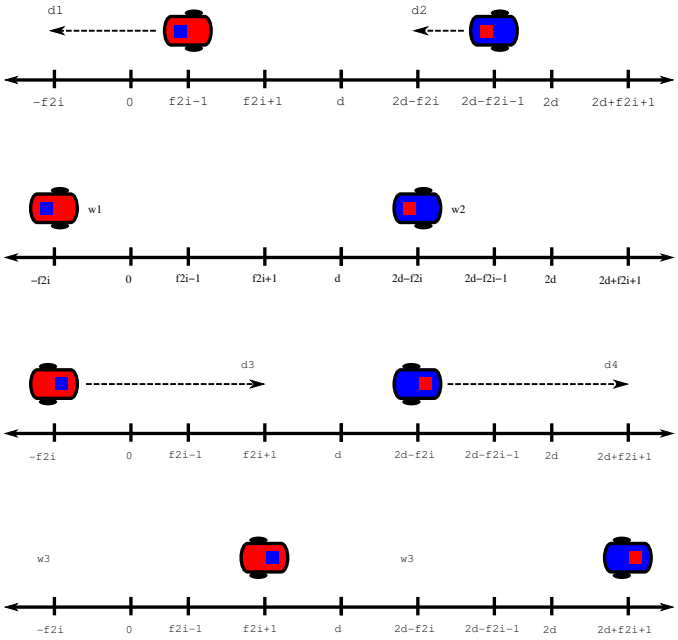


Fig. 1. Behavior in round i. The robots start in configuration $(f_{2i+1}, 2d - f_{2i-1})$. The current coin flips require both robots to move left-then-right in the current round. **FIRST:** First phase movements. Robot-1 moves left $d_1 = r^{2i+\epsilon_1} + r^{2i-1+\epsilon_1}$. Robot-2 moves left $d_2 = r^{2i+\epsilon_2} - r^{2i-1+\epsilon_2}$. **SECOND:** First phase idle time. To ensure synchronous rounds, robot-j waits $r^{2i} + r^{2i+1} - d_j$ time units before starting the next phase. **THIRD:** Second phase movement. Robot-1 travels right $d_3 = r^{2i+\epsilon_1} + r^{2i+1+\epsilon_1}$ and robot-2 travels right $d_4 = r^{2i+\epsilon_2} + r^{2i+1+\epsilon_2}$. **FOURTH:** Second phase idle time. Robot j waits $r^{2i+1} + r^{2i+2} - d_j$.

If we were to choose $r = 2$ (as in Spiral LostCow), our symmetric rendezvous algorithm would yield an unbounded competitive ratio. Intuitively, the growth of the distance traveled overwhelms the probability of the rendezvous of the robots. Instead of doubling r, our algorithm increases the reach f_i of the robots by a factor $r \approx 1.195$ (the derivation of this optimized value is presented in the proof of Theorem 1). Given the initial distance $2d$, set $\delta \in (0, 1]$ and $k \in \mathbb{Z}^+$ to be the unique numbers achieving $d = r^{k+\delta}$.

The algorithm proceeds in rounds, indexed by integers $i \geq 0$ (starting at $i = 0$ gives a more natural indexing). In each round, the robot will alternate between moving and waiting. A round is *successful* if the robots achieve rendezvous in that round; otherwise it is *unsuccessful*. We describe the movement of robot-1, who starts at $x = 0$. At each round i, the robot flips a coin to determine its itinerary (right-then-left or left-then-right) for this round. We divide each round into two phases, according to the direction of movement. In the ith round, the robot starts at one of $x = \pm f_{2i-1}$, each with probability $1/2$. If the robot tosses heads, then in phase 1, it moves right to the point $x = f_{2i}$ and in phase 2, it moves left to the point $x = -f_{2i+1}$. If the robot tosses tails, it moves left to $x = -f_{2i}$ in phase 1 and then right to $x = f_{2i+1}$ in phase 2. At the end

Algorithm 1 $\mathcal{SR}$: Symmetric rendezvous strategy on the line for unknown initial distance.

```
r ← 1.195
ε ← random from (0, 1]
  {Note that f(j) = r^{j+ε} below}
i ← 0
coinFlip ← random from {−1, 1}
previousCoinFlip ← coinFlip
while checkRendezvous() != true do
   if i = 0 then
      D_{i,1} ← f(2i)
   else if coinFlip != previousCoinFlip then
      D_{i,1} ← f(2i) − f(2i − 1)
   else
      D_{i,1} ← f(2i − 1) + f(2i)
   end if
   W_{i,1} ← f(2i) + f(2i + 1) − D_{i,1}
   moveTo(coinFlip * f(2i))
   wait(W_{i,1})
   D_{i,2} ← f(2i) + f(2i + 1)
   W_{i,2} ← f(2i + 1) + f(2i + 2) − D_{i,2}
   moveTo(−1 * coinFlip * f(2i + 1))
   wait(W_{i,2})
   i ← i + 1
   previousCoinFlip ← coinFlip
   coinFlip ← random from {−1, 1}
end while
```

of an unsuccessful round $i \geq 0$, the possible configurations of the robots are $(\pm f_{2i+1}, 2d \pm f_{2i+1})$. This is also the initial configuration for round $i+1$.

Let $D_{i,1}$ ($D_{i,2}$) denote the distance traveled in the first (second) phase of round i, and let $D_i = D_{i,1} + D_{i,2}$ denote the total distance traveled in the round. The value of $D_{i,1}$ depends upon whether the coin flip in this round differs from the coin flip in the previous round. If they differ, then $D_{i,1} = f_{2i} - f_{2i-1}$. If they match, then $D_{i,1} = f_{2i} + f_{2i-1}$. (Note that these statements are also true for $i = 0$ because $f_{-1} = 0$.) Regardless of the coin flip, $D_{i,2} = f_{2i+1} + f_{2i}$. The distance traveled (the length of an itinerary) by a robot in an unsuccessful round i is either $D_i = f_{2i+1} + 2f_{2i} - f_{2i-1}$ or $D_i = f_{2i+1} + 2f_{2i} + f_{2i-1}$, each with equal probability.

Recall that the robots use different ϵ values for their movements. In order to keep the robot motions in sync, we introduce wait times after each movement (left or right). A robot must assume the worst: that the other robot is using $\epsilon = 1$, and therefore requires time $T_{i,1} = r^{2i} + r^{2i+1}$ in phase 1 and time $T_{i,2} = r^{2i+1} + r^{2i+2}$ in phase 2. After each movement, the robot will idle long enough to allow the other robot to complete its current motion. Keeping robot motions in sync guarantees that, once their displacement is large enough, the probability that the robots meet is 1/2.

We summarize the activity of the robot in round i. The robot flips a coin. In phase 1, it moves distance $D_{i,1}$ and waits for time $W_{i,1} = T_{i,1} - D_{i,1}$. In phase 2, it moves distance $D_{i,2}$ and waits for time $W_{i,2} = T_{i,2} - D_{i,2}$. Assuming that the round is unsuccessful, the time elapsed in round i is $T_i = T_{i,1} + T_{i,2}$, independent of both the robot's ϵ value and the coin flip. The algorithm terminates the first time that the robot configuration is of the form (x, x). The pseudocode of the strategy is presented in Algorithm 1. Our main result is the following theorem.

Theorem 1: The optimal expansion radius for the symmetric rendezvous algorithm is $r \approx 1.195$. This choice of r gives an algorithm that is 17.686-distance competitive and 24.843-time competitive.

We prove this theorem in the next subsection.

C. The Analysis

Performance analysis of Algorithm $\mathcal{SR}$ for $d = r^{k+\delta}$ depends upon the parity of k. The even case is slightly more difficult to analyze, so we work through that analysis for the distance competitive ratio. The remaining analyses (even time, odd distance and odd time) proceed similarly. These analyzes can be found in the accompanying technical report [19].

Henceforth, we assume that k is even. We divide the execution of Algorithm $\mathcal{SR}$ into four stages. *Stage-1* is an initialization stage whose length depends on the (unknown) initial distance $2d = 2r^{k+\delta}$. This stage encompasses the early rounds $0 \leq i \leq k/2 - 1$ in which the robots do not move far enough to meet. The first round in which the robots might meet is the $k/2$-th round. *Stage-2* consists of only this critical round. We define a separate *Stage-3* for round $k/2 + 1$ as well. The initial locations of the robots exhibit a transient behavior in this round. *Stage-4* contains all rounds after Stage-3. In these later rounds, the initial configurations and the rendezvous behavior are consistent, so we can compute the expected distance traveled using an infinite sum.

In our analysis, we track the direction of movement (left-then-right or right-then-left) of each robot. In Stage-2, we also consider the relative size of δ compared to ϵ_1, ϵ_2. In Stage-3, we must pay close attention to the possible initial configurations. Note that, starting at Stage-2, it is also necessary to calculate the distance traveled at each round based on whether the robots meet at this round or not.

We now introduce the variables used in our analysis. Let R_i be the event that the algorithm is still active in round i. Assuming that R_i holds, we define the following: S_i is the event that the robots successfully meet in round i; A_i is the event that robot-1 initially moves to the right (towards robot-2) in round i; B_i is the event that robot-2 initially moves to the left (towards robot-1) in round i.

Due to the symmetric strategies, the expected distance traveled is identical for both robots. Therefore, we only analyze robot-1. Still assuming that R_i holds, we again define $D_i = D_{i,1} + D_{i,2}$ and $T_i = T_{i,1} + T_{i,2}$ and $W_i = W_{i,1} + W_{i,2}$.

The algorithm is always active in the first round, thus $\mathbb{P}[R_0] = 1$. The probability that the algorithm is still active in round i is the joint probability of the events that the robots do not meet in the rounds up to round i. That is, $\mathbb{P}[R_i] = \mathbb{P}[\overline{S}_0 \wedge \cdots \wedge \overline{S}_{i-1}]$ for $i > 0$. Restating our

observations from the previous section, we have

$$\begin{aligned}\mathbb{E}[D_i \mid \overline{S}_i] &= \mathbb{E}[f_{2i+1} + 2f_{2i} \mid \overline{S}_i] \\ &= (r^{2i+1} + 2r^{2i})\mathbb{E}[r^\epsilon \mid \overline{S}_i], \quad (2)\\ \mathbb{E}[T_i \mid \overline{S}_i] &= r^{2i+2} + 2r^{2i+1} + r^{2i}, \quad (3)\\ \mathbb{E}[W_i \mid \overline{S}_i] &= \mathbb{E}[T_i \mid \overline{S}_i] - \mathbb{E}[D_i \mid \overline{S}_i]. \quad (4)\end{aligned}$$

During Stage-1, we have $\mathbb{E}[r^\epsilon \mid \overline{S}_i] = \mathbb{E}[r^\epsilon]$. During the later stages, (i.e. when $i \geq k/2$), the size of ϵ and the event S_i are dependent variables.

We are now ready to study the four stages of Algorithm $\mathcal{SR}$ for even k. Sections III-C1, III-C2, III-C3 and III-C4 present the distance analysis of stages 1–4 respectively. For clarity of exposition, we defer evaluation of certain probabilities and expectations until Section III-C5, where we prove the distance competitive ratio for even k in our main theorem.

*1) **Analysis of Stage-1 for even** k:* Recall that $d = r^{k+\delta}$ where $\delta \in (0, 1]$. We compute the expected distance traveled during Stage-1. When $0 \leq i \leq k/2 - 1$, the combined robot displacement is

$$r^{2i+\epsilon_1} + r^{2i+\epsilon_2} \leq 2r^{2i+2} \leq 2r^k < 2r^{k+\delta} = d,$$

so the robots cannot meet during Stage-1. In other words, $\mathbb{P}[R_i] = 1 = \mathbb{P}[\overline{S}_i]$ for $0 \leq i \leq k/2 - 1$.

Lemma 2: The expected distance traveled during Stage-1 satisfies

$$\sum_{i=0}^{k/2-1} \mathbb{E}[D_i \mid R_i]\mathbb{P}[R_i] \quad < \quad (r+2)\,\frac{r^k}{r^2-1}\,\mathbb{E}[r^\epsilon].$$

Furthermore, the four final configurations $(\pm f_{k-1}, 2d \pm f_{k-1})$ each occur with probability $1/4$.

Proof. By equation (2),

$$\begin{aligned}\sum_{i=0}^{k/2-1} \mathbb{E}[D_i \mid R_i]\mathbb{P}[R_i] &= \sum_{i=0}^{k/2-1} \mathbb{E}\left[D_i \mid \overline{S}_i\right]\mathbb{P}\left[\overline{S}_i\right] \\ &= \sum_{i=0}^{k/2-1} (r^{2i+1} + 2r^{2i})\mathbb{E}\left[r^\epsilon \mid \overline{S}_i\right] \cdot 1 \\ &= (r+2)\mathbb{E}\left[r^\epsilon\right] \sum_{i=0}^{k/2-1} r^{2i} = (r+2)\,\frac{r^k-1}{r^2-1}\,\mathbb{E}[r^\epsilon].\end{aligned}$$

Considering round $k/2 - 1$, clearly each of the four final configurations $(\pm f_{k-1}, 2d \pm f_{k-1})$ are equiprobable. □

*2) **Analysis of Stage-2 for even** k:* We bound the expected distance traveled by a robot in the critical round, $k/2$. Unlike Stage-1, the robots meet with nonzero probability. We separately calculate the distance traveled in a successful and an unsuccessful $k/2$-th round in Lemmas 4 and 5 respectively. Adding these expressions gives the total expected distance traveled in the critical round.

The robots cannot meet prior to the critical round, so $\mathbb{P}[R_{k/2}] = 1$. Furthermore, $S_{k/2} \wedge R_{k/2} = S_{k/2}$ and $\overline{S}_{k/2} \wedge R_{k/2} = \overline{S}_{k/2}$. We now capture the success and failure conditions for the critical round. Define C to be the event that $2\delta \leq \epsilon_1 + \epsilon_2$. We claim that the robots successfully rendezvous in the first phase of round $k/2$ when $A_k \wedge B_k \wedge C$ holds. Indeed, the first phase of round $k/2$ results in rendezvous if and only if $2r^\delta \leq r^{\epsilon_1} + r^{\epsilon_2}$, or equivalently $\epsilon_2 \geq \log_r(2r^\delta - r^{\epsilon_1})$. This function is concave down for $0 \leq \epsilon_1 \leq 1$, so the function lies below the tangent line through the point (δ, δ). The formula for this tangent line is $\epsilon_2 = 2\delta - \epsilon_1$. Therefore, if $2\delta \leq \epsilon_1 + \epsilon_2$ then $2r^\delta \leq r^{\epsilon_1} + r^{\epsilon_2}$.

For simplicity of analysis, we use the tangent line $\epsilon_2 = 2\delta - \epsilon_1$ to approximate the function $\epsilon_2 = \log_r(2r^\delta - r^{\epsilon_1})$ on the unit square. The tangent line lies above the curve in this region. Therefore, we use the event C as a proxy for success when $A_{k/2} \wedge B_{k/2}$ holds. This yields an upper bound on the expected distance traveled during the algorithm (because we mis-categorize some rendezvous successes in round $k/2$ as failures). In calculations below, we find that the optimal expansion radius satisfies $1 \leq r \leq \sqrt{2}$. For these r values, the error introduced is minor compared to the other factors.

With this in mind, when the event $A_{k/2} \wedge B_{k/2}$ holds, we *define* success to coincide with event C. This results in an upper bound on the competitive ratio, due to the resulting approximation. We note that the event C will continue to play a role in our analysis of subsequent stages.

Lemma 3: The probability that the critical round is successful is

$$\mathbb{P}[S_{k/2}] = (1 + \mathbb{P}[C])/4.$$

In addition, the ending configuration $x = (x_1, x_2)$ for an unsuccessful critical round satisfies

$$\begin{aligned}\mathbb{P}[x = (-f_{k+1}, 2d - f_{k+1}) \mid \overline{S}_{k/2}] &= 1/4,\\ \mathbb{P}[x = (f_{k+1}, 2d + f_{k+1}) \mid \overline{S}_{k/2}] &= 1/4,\\ \mathbb{P}[x = (-f_{k+1}, 2d + f_{k+1}) \mid \overline{S}_{k/2}] &= \mathbb{P}[\overline{C}]/4.\end{aligned}$$

Proof. The robots can rendezvous in two ways. The first way is when they both move away from each other in Phase-1 of the round. Call this event

$$E_1 = \overline{A}_{k/2} \wedge\ \overline{B}_{k/2}.$$

The second way is when the event

$$E_2\ = A_{k/2} \wedge B_{k/2} \wedge C$$

holds. We have $\mathbb{P}[E_1 \vee E_2] = \mathbb{P}[E_1] + \mathbb{P}[E_2] = 1/4 + \mathbb{P}[C]/4$.

An unsuccessful round corresponds to one of the events $A_k \wedge \overline{B}_k$, $\overline{A}_k \wedge B_k$ or $A_k \wedge B_k \wedge \overline{C}$. The ending configurations and probabilities listed in the lemma correspond to these three events. □

Before proceeding further, we make one important remark: the occurrence of event C depends upon the (unknown) value of the input variable δ. Specifically, the formulas for $\mathbb{P}[C]$, $\mathbb{P}[\overline{C}]$, and $\mathbb{E}[r^\epsilon \mid C]$, $\mathbb{E}[r^\epsilon \mid \overline{C}]$ depend on whether δ is greater or less than $1/2$. Calculations concerning C are found in Appendix A. During the final analysis of Theorem 1, the competitive ratio of our algorithm will be computed by choosing the δ value which gives the maximum ratio. This dependence on the input variable δ is one of the reasons that we defer evaluation of certain probabilities and expectations until Section III-C5

Lemma 4: The expected distance traveled in a successful $k/2$-th round is

$$\mathbb{E}[D_{k/2} \mid S_{k/2}]\mathbb{P}[S_{k/2}] = \frac{1}{2}r^k\mathbb{E}[r^\epsilon] + d\left(\frac{1+\mathbb{P}[C]}{4}\right).$$

Proof. We use the same notation E_1, E_2 found in the proof of Lemma 3. The expected distance traveled in a successful $k/2$-th round is given by

$$\begin{aligned}&\mathbb{E}\left[D_{k/2}|S_{k/2}\right]\mathbb{P}\left[S_{k/2}\right]\\ &= \mathbb{E}\left[D_{k/2}|E_1\right]\mathbb{P}\left[E_1\right] + \mathbb{E}\left[D_{k/2}|E_2\right]\mathbb{P}\left[E_2\right],\end{aligned} \tag{5}$$

where $\mathbb{P}[E_1] = 1/4$ and $\mathbb{P}[E_2] = \mathbb{P}[C]/4$.

Next we compute the expected distance traveled for each of the events E_1 and E_2. The four equiprobable initial configurations in this round are $(\pm f_{k-1}, 2d \pm f_{k-1})$. Calculations similar to equation (2), give

$$\mathbb{E}[D_{k/2} \mid E_1] = \mathbb{E}[2f_k + d \mid E_1] = 2r^k\mathbb{E}\left[r^{\epsilon_1}\right] + d.$$

We also have

$$\begin{aligned}&\mathbb{E}[D_{k/2} \mid E_2]\\ &= \mathbb{E}\left[\frac{1}{4}\left(2(d-f_{k-1}) + 2(d+f_{k-1})\right)\right] = d.\end{aligned}$$

Using these values in equation (5) completes the proof. □

Lemma 5: The expected distance traveled in an unsuccessful $k/2$-th round is

$$\begin{aligned}&\mathbb{E}\left[D_{k/2}|\overline{S}_{k/2}\right]\mathbb{P}\left[\overline{S}_{k/2}\right]\\ &= r^k(2+r)\left(\frac{1}{2}\mathbb{E}\left[r^{\epsilon_1}\right] + \frac{1}{4}\mathbb{E}\left[r^{\epsilon_1}|\overline{C}\right]\mathbb{P}[\overline{C}]\right).\end{aligned}$$

Proof. In this round, the robots cannot meet under two conditions. First, they do not meet when they move in tandem. In other words the event

$$E_3 = \left(\overline{A}_{k/2} \wedge B_{k/2}\right) \vee \left(A_{k/2} \ \wedge \ \overline{B}_{k/2}\right)$$

holds. Secondly, if they move towards each other and ϵ_1, ϵ_2 are not large enough for them to meet. In other words, the event

$$E_4 \ = \ A_{k/2} \wedge \ B_{k/2} \wedge \ \overline{C}$$

holds. The expected distance traveled in an unsuccessful $k/2$-th round is given by

$$\begin{aligned}&\mathbb{E}\left[D_{k/2}|\overline{S}_{k/2}\right]\mathbb{P}\left[\overline{S}_{k/2}\right]\\ &= \mathbb{E}\left[D_{k/2}|E_3\right]\mathbb{P}\left[E_3\right] + \mathbb{E}\left[D_{k/2}|E_4\right]\mathbb{P}\left[E_4\right].\end{aligned} \tag{6}$$

We have $\mathbb{P}[E_3] = 1/2$ and $\mathbb{P}[E_4] = \mathbb{P}[\overline{C}]/4$. Next we compute the expected distance traveled for each of the events E_3, E_4. Similar to equation (2), these expected distances are

$$\begin{aligned}\mathbb{E}[D_{k/2}|E_3] &= r^k(2+r)\mathbb{E}\left[r^{\epsilon_1}\right],\\ \mathbb{E}[D_{k/2}|E_4] &= r^k(2+r)\mathbb{E}\left[r^{\epsilon_1}|\overline{C}\right].\end{aligned}$$

Using these values in equation (6) proves the lemma. □

*3) **Analysis of Stage-3 for even** k:* We compute the expected distance traveled by a robot in the transitional $(k/2+1)$-th round. Assuming that the robots did not meet in Stage-2, the robots meet in Stage-3 if and only if

$$E_5 = (A_{k/2+1} \wedge B_{k/2+1}) \vee (\overline{A}_{k/2+1} \wedge \overline{B}_{k/2+1})$$

holds. Therefore we have $\mathbb{P}[S_{k/2+1} \mid R_{k/2+1}] = 1/2 = \mathbb{P}[\overline{S}_{k/2+1} \mid R_{k/2+1}]$. We continue to use the event notation from the previous section. We enter Stage-3 directly after event E_3 or E_4, and we handle these cases separately.

Lemma 6: Suppose that we enter round $k/2+1$ after event E_3. We have

$$\mathbb{E}[D_{k/2+1} \mid S_{k/2+1} \wedge E_3] = r^{k+2}\mathbb{E}[r^\epsilon] + d$$

and

$$\mathbb{E}[D_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_3] = r^{k+2}(2+r)\mathbb{E}[r^\epsilon].$$

Furthermore, when $\overline{S}_{k/2+1}$ occurs, the ending configuration is either $(f_{k+3}, 2d+f_{k+3})$ or $(-f_{k+3}, 2d-f_{k+3})$, each with probability $1/2$.

Proof. These values are calculated analogously to equation (2), using the equivalence $S_{k/2+1} = E_5$. For both events $S_{k/2+1}$ $\overline{S}_{k/2+1}$, there are two initial configurations, and two possible itineraries. Averaging over the four equiprobable events gives the lemma. □

When E_4 has just occurred, the round $k/2+1$ starts at position $(-f_{k+1}, 2d + f_{k+1})$. The proof of the following lemma is analogous to that of Lemma 6.

Lemma 7: Suppose that we enter round $k/2+1$ after event E_4. We have $\mathbb{E}[D_{k/2+1} \mid E_4] = \frac{1}{2}\mathbb{E}[D_{k/2+1} \mid S_{k/2+1} \wedge E_4] + \frac{1}{2}\mathbb{E}[D_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_4]$ where

$$\mathbb{E}[D_{k/2+1} \mid S_{k/2+1} \wedge E_4] = r^{k+2}\mathbb{E}[r^\epsilon \mid \overline{C}] + d$$

and

$$\mathbb{E}[D_{k/2+1} \mid \overline{S}_{k/2+1} \wedge E_4] = r^{k+2}(2+r)\mathbb{E}[r^\epsilon \mid \overline{C}].$$

Furthermore, when $\overline{S}_{k/2+1}$ occurs, the ending configuration is either $(f_{k+3}, 2d+f_{k+3})$ or $(-f_{k+3}, 2d-f_{k+3})$, each with probability $1/2$. □

We now combine all our terms into a single expression.

Lemma 8: The expected distance traveled in round $k/2+1$ is

$$\begin{aligned}&\mathbb{E}[D_{k/2+1} \mid R_{k/2+1}]\mathbb{P}[R_{k/2+1}]\\ &= \mathbb{E}[D_{k/2+1} \mid E_3]\mathbb{P}[E_3] + \mathbb{E}[D_{k/2+1} \mid E_4]\mathbb{P}[E_4]\end{aligned}$$

where

$$\mathbb{E}[D_{k/2+1} \mid E_3]\mathbb{P}[E_3] = \frac{1}{4}\left(d + r^{k+2}(3+r)\mathbb{E}[r^\epsilon]\right)$$

and

$$\begin{aligned}&\mathbb{E}[D_{k/2+1} \mid E_4]\mathbb{P}[E_4]\\ &= \frac{\mathbb{P}[\overline{C}]}{8}\left(d + r^{k+2}(3+r)\mathbb{E}\left[r^\epsilon \mid \overline{C}\right]\right).\end{aligned}$$

Proof. The probability of success in round $k/2+1$ is always $1/2$, and we know that $\mathbb{P}[E_3] = 1/2$ and $\mathbb{P}[E_4] = \mathbb{P}[\overline{C}]/4$. The result follows from Lemmas 6 and 7. □

*4) **Analysis of Stage-4 for even** k:* We compute the expected distance traveled for all rounds $i \geq (k/2+2)$. The events E_3 and E_4 still influence the expected values in these later rounds. In particular, for $i \geq k/2+2$, we have

$$\begin{aligned}
&\mathbb{E}[r^\epsilon \mid R_i]\mathbb{P}[R_i] \\
&= \mathbb{E}[r^\epsilon \mid R_i \wedge E_3]\mathbb{P}[R_i \mid E_3]\mathbb{P}[E_3] \\
&\quad +\mathbb{E}[r^\epsilon \mid R_i \wedge E_4]\mathbb{P}[R_i \mid E_4]\mathbb{P}[E_4] \\
&= \mathbb{E}[r^\epsilon]\mathbb{P}[R_i \mid E_3]\mathbb{P}[E_3] \\
&\quad +\mathbb{E}[r^\epsilon \mid \overline{C}]\mathbb{P}[R_i \mid E_4]\mathbb{P}[E_4]. \qquad (7)
\end{aligned}$$

Note that if E_4 holds, then so does $\overline{C}$, which in turn influences the expected value of r^ϵ. However, the rendezvous behavior of round $i \geq k/2+2$ is independent of how we got there: we always achieve rendezvous with probability $1/2$. This behavior is summarized in the following lemma.

Lemma 9: For $i \geq k/2+2$, we have

$$\mathbb{P}[R_i \mid E_3] = \mathbb{P}[R_i \mid E_4] = \left(\frac{1}{2}\right)^{i-k/2-1}.$$

Furthermore,

$$\begin{aligned}
\mathbb{E}[D_i \mid R_i \wedge E_3] &= \frac{1}{2}\left(d + r^{2i}\,(3+r)\,\mathbb{E}[r^\epsilon]\right), \\
\mathbb{E}[D_i \mid R_i \wedge E_4] &= \frac{1}{2}\left(d + r^{2i}\,(3+r)\,\mathbb{E}[r^\epsilon \mid \overline{C}]\right).
\end{aligned}$$

Proof. We prove the lemma given E_3 occurred in the critical round $k/2$. The proof for E_4 is analogous.

The probability that round $k/2+1$ was unsuccessful is $1/2$. Likewise, every round $i \geq k/2+2$ is successful half the time, namely when $(A_i \wedge B_i) \vee (\overline{A}_i \wedge \overline{B}_i)$ holds. This proves the first statement for E_3. Considering the expected distance traveled in round i, we find that

$$\begin{aligned}
\mathbb{E}[D_i \mid R_i \wedge E_3] &= \mathbb{E}[D_i \mid R_i \wedge E_3 \wedge S_i]\mathbb{P}[S_i] \\
&\quad +\mathbb{E}[D_i \mid R_i \wedge E_3 \wedge \overline{S}_i]\mathbb{P}[\overline{S}_i] \\
&= \frac{1}{2}\left(r^{2i}\mathbb{E}[r^\epsilon \mid E_3] + d\right) + \frac{1}{2}\left(r^{2i}(2+r)\mathbb{E}[r^\epsilon \mid E_3]\right) \\
&= \frac{1}{2}\left(d + r^{2i}(3+r)\mathbb{E}[r^\epsilon]\right).
\end{aligned}$$

This concludes the proof for E_3. The only difference in the proofs for E_4 is the fact that $\mathbb{E}[r^\epsilon \mid E_4] = \mathbb{E}[r^\epsilon \mid \overline{C}]$. □

We now compute the expected distance traveled in Stage-4.

Lemma 10: The expected distance traveled in Stage-4 is

$$\begin{aligned}
&\sum_{i=k/2+2}^{\infty} \mathbb{E}[D_i \mid R_i]\mathbb{P}[R_i] \\
&= \frac{1}{4}\left(d + \frac{r^{k+4}(3+r)}{2-r^2}\mathbb{E}[r^\epsilon]\right) \\
&\quad +\frac{\mathbb{P}[\overline{C}]}{8}\left(d + \frac{r^{k+4}(3+r)}{2-r^2}\mathbb{E}[r^\epsilon \mid \overline{C}]\right).
\end{aligned}$$

Proof. We split our calculation according to $R_i \wedge E_3$ and $R_i \wedge E_4$. If $R_i \wedge E_3$ holds, then by Lemma 9 we have

$$\begin{aligned}
&\sum_{i=k/2+2}^{\infty} \mathbb{E}[D_i \mid R_i \wedge E_3]\mathbb{P}[R_i \mid E_3]\mathbb{P}[E_3] \\
&= \sum_{i=k/2+2}^{\infty} \frac{1}{2}\left(d + r^{2i}(3+r)\mathbb{E}[r^\epsilon]\right)\left(\frac{1}{2}\right)^{i-k/2-1}\left(\frac{1}{2}\right) \\
&= \frac{1}{4}\sum_{j=0}^{\infty}\left(d + r^{2j+k+4}(3+r)\mathbb{E}[r^\epsilon]\right)\left(\frac{1}{2}\right)^{j+1} \\
&= \frac{d}{8}\sum_{j=0}^{\infty}\left(\frac{1}{2}\right)^j + \frac{r^{k+4}(3+r)}{8}\mathbb{E}[r^\epsilon]\sum_{j=0}^{\infty}\left(\frac{r^2}{2}\right)^j \\
&= \frac{1}{4}\left(d + \frac{r^{k+4}(3+r)}{2-r^2}\mathbb{E}[r^\epsilon]\right).
\end{aligned}$$

The calculations for $R_i \wedge E_4$ are entirely analogous, and give

$$\begin{aligned}
&\sum_{i=k/2+2}^{\infty} \mathbb{E}[D_i \mid R_i \wedge E_4]\mathbb{P}[R_i \wedge E_4]\mathbb{P}[E_4] \\
&= \frac{\mathbb{P}[\overline{C}]}{8}\left(d + \frac{r^{k+4}(3+r)}{2-r^2}\mathbb{E}[r^\epsilon \mid \overline{C}]\right).
\end{aligned}$$

□

5) **Computing the competitive ratio of SR***:* Having determined the expected distance traveled in all stages, we can now prove our main result. We show that the optimal expansion radius is $r \approx 1.195$ which guarantees an algorithm that is 17.686-distance competitive and 24.843-distance competitive.

Proof of Theorem 1. We sketch the proof for the distance competitive ratio for even k. Additional calculations are available in [19]. The total expected distance traveled is obtained by adding the expressions in Lemmas 2, 4, 5, 8 and 10. This expression is quite long, so we omit its formulation. Recalling that $d = r^{k+\delta}$, we first replace each occurrence of r^k with $d\,r^{-\delta}$. Then we divide by d to obtain the competitive ratio.

Next we evaluate the probabilities $\mathbb{P}[C]$, $\mathbb{P}[\overline{C}]$ and the expectations $\mathbb{E}[r^\epsilon]$, $\mathbb{E}[r^\epsilon \mid C]\mathbb{P}[C]$ and $\mathbb{E}[r^\epsilon \mid \overline{C}]\mathbb{P}[\overline{C}]$, as found in Appendix A. We note that the values involving C have a dependence on the value of the input variable δ. Specifically, we must perform two sets of calculations, depending on whether $0 < \delta \leq 1/2$ or $1/2 < \delta \leq 1$. For each, we will choose the δ which maximizes the distance competitive ratio.

For $0 < \delta \leq 1/2$, this expression is maximized at $\delta = 1/2$. In turn, the choice of $r \approx 1.195$ gives the minimum competitive ratio guarantee of 17.48. Next, the expression for $1/2 < \delta \leq 1$ is maximized for $\delta = 1$. The choice of $r \approx 1.195$ also minimizes this function. This value gives a competitive ratio guarantee of 17.686. Therefore, for any input distance d, our algorithm guarantees a competitive ratio of 17.686.

The time analysis for even k is analogous to the above calculations. It gives a time competitive ratio of 24.85. A sketch of this analysis is given in [19]. The time and distance analysis for odd k is also quite similar to the above calculations. The details are sketched in [19], where we show that the odd distance competitive ratio is 17.686 and the odd time competitive ratio is 23.58. □

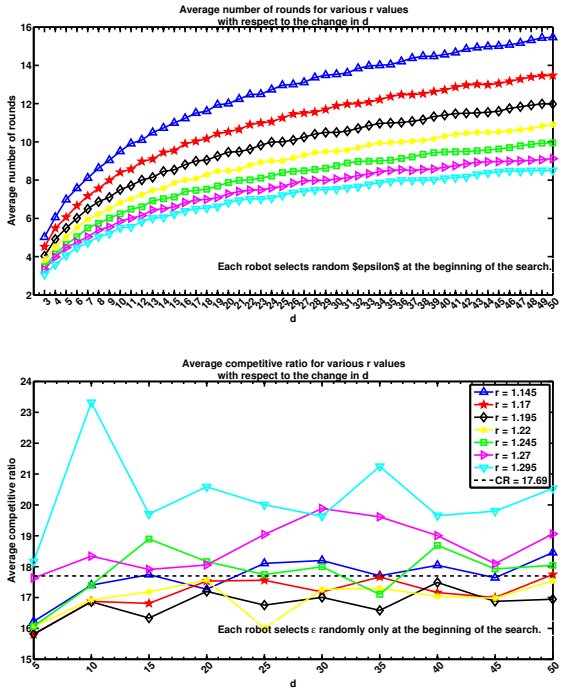


Fig. 2. The average performance of algorithm $\mathcal{SR}$ in $10,000$ simulations. Here, ϵ is randomly chosen only at the beginning of the search. **TOP:** Average number of rounds with respect to the change in the initial distance for various r values. **BOTTOM:** Competitive ratio for the average distance traveled with respect to the change in d for various r values.

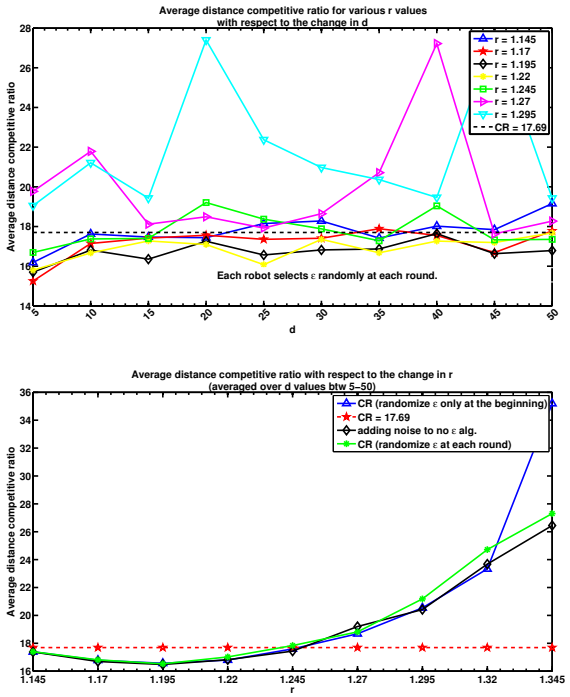


Fig. 3. The performance of our algorithm $\mathcal{SR}$ in simulations when each robot generates a new ϵ value for each round. **TOP:** Competitive ratio for the average distance traveled with respect to the change in d for various r values. **BOTTOM:** The effect of Gaussian noise on the competitive ratio of our algorithm. The distance competitive ratio is averaged over various initial distance values for each r value on the x-axis.

IV. Simulations

In this section, we investigate the performance of the Algorithm $\mathcal{SR}$ in simulations. So far we assumed that the robots start at the same time. Although this is a standard assumption, it may be violated in practice. Therefore, we divide the simulations into two groups with respect to the time the robots start executing the algorithm: the synchronous case (same starting time) and the asynchronous case (different starting times). Sections IV-A and IV-B respectively present the simulation results for the first and the second group.

A. Synchronous Starting Times

In this section, we show that the simulation results of the synchronous case of $\mathcal{SR}$ agree with the optimal choice of r and the competitive ratio we obtained in Theorem 1. Moreover, we also show that randomizing ϵ at each round is analogous to errors proportional to distance. This yields the observation that our algorithm is robust to navigational errors.

First we investigate the performance of $\mathcal{SR}$ as a function of r and initial distance. The plots in Figure 2 report the results of simulations for various r values and initial distances, each averaged over $10,000$ trials. The initial distance between the robots is $2d$. We simulate our strategy for integer d values varied between 5 and 50 step sizes and the robot has a speed of unit step size. The value of r is started at 1.145 and incremented by 0.025 up to 1.295.

The top plot shows the change in average number of rounds for these r values, as a function of initial distance. As expected, the simulation results show that the average number of rounds increases as the initial distance increases and as the value of r decreases. Indeed, the average number of rounds which is $\approx (k/2+1)$ is highest when $r = 1.145$ and lowest when $r = 1.295$.

The bottom plot in Figure 2 shows the average competitive ratio as a function of initial distance for the same set of r values. The horizontal dotted line shows the theoretically optimal distance competitive ratio of 17.696 for the optimal value of $r = 1.195$. The black solid line shows the performance of $\mathcal{SR}$ in simulations for $r = 1.195$. In these simulations, we observe that $r = 1.195$ has superior performance for most initial distance values.

Next, we consider a variant of the $\mathcal{SR}$ algorithm in which a new value of $\epsilon \in (0, 1]$ is generated in each round. The effect of this randomness can be interpreted as navigational errors, for example due to odometry errors. The performance of this variant shows that our algorithm is robust under such errors. First, we simulate these errors by selecting $\epsilon \in (0, 1]$ uniformly random for each robot at each round. Analysis similar to that of Theorem 1 shows that this variant has the same expected competitive ratio. The results of our simulations, found in Figure 3, agree with this claim. The top plot shows the average distance competitive ratio for various r values, plotted as a function of initial distance. The results are comparable to those in Figure 2.

In the literature, the noise caused by navigational errors is often modeled as a Gaussian whose standard deviation is

proportional to the distance traveled [20]. In the bottom plot in Figure 3, we execute $\mathcal{SR}$ in the presence of such errors. In these simulations, we assume that the errors occur only in the X-axis. We use $\mu = -1.843$ and $\sigma = 0.372$ from Table 1 of [20]. Specifically, rather than taking $f_i = r^{i+\epsilon}$, we set $f_i = r^i + n(\mu, \sigma)$ where n is the Gaussian noise. These values are for a robot moving at 0.2 m/sec.

The horizontal dotted line shows the $\mathcal{SR}$'s distance competitive ratio of 17.696 for the optimal value $r = 1.195$. The competitive ratio for radius r is the average of the observed competitive ratio over all simulated d values. In other words, we ran $10,000$ simulations for each integer distances $5 \leq d \leq 50$, and then averaged their observed competitive ratios. The black solid line shows the change in the competitive ratio of $\mathcal{SR}$ with respect to the change in r. The red solid line shows the change in the competitive ratio when Gaussian noise is added to $\mathcal{SR}$. The simulation results indicate that our algorithm is robust to navigational errors. Indeed, for most of the r values appear in the x-axis, Gaussian noise does not effect the performance our algorithm. This means that adding the random variable ϵ in f_i is nearly identical to running $\mathcal{SR}$ with noise.

B. Asynchronous Starting Times

Our main contribution in this paper is the rendezvous strategy $\mathcal{SR}$ and its performance analysis under the assumption that the robots that start executing their strategies simultaneously. This synchronized start requires a global control of the robots which may not be always possible in practice. In this section, we study the performance of the algorithm $\mathcal{SR}$ when this assumption is violated. We consider simulations in which robot-2 starts t time units later than robot-1. Figure 4 shows the average competitive ratio for distances $5 \leq d \leq 1000$ and time delays $0 \leq t \leq 1000$. These simulation results suggest that our algorithm is robust under synchronization errors. Very small time delays have little impact on performance. For larger initial distance d, the distance competitive ratio stays below 22. Furthermore, very long delays actually improve performance. Indeed, we expect this behavior since our symmetric rendezvous problem becomes closer to the asymmetric case, where one robot "waits for mommy." In other words, for very large t (compared to d), we have a lost cow problem. We see a clear downward trend on the competitive ratio as t increases. Indeed, for smaller d values, the competitive ratio stabilizes at a value just above 10.

Continuing along this line, if we anticipate major synchronization errors, then our algorithm should be adapted to better fit the scenario. The idle times of $\mathcal{SR}$ are designed precisely to keep our robot motions synchronized. If a near-synchronous start time is not guaranteed, these idles times are ineffective. Therefore, we considered a second set of simulations in which we eliminate the idle times from the $\mathcal{SR}$ algorithm. We call the resulting algorithm $\mathcal{ASR}$, for *asynchronous symmetric rendezvous*.

We studied the performance of $\mathcal{ASR}$ in two phases. First, we empirically determined a good choice of r for the asynchronous case. Next, we studied the performance of $\mathcal{ASR}$ as a

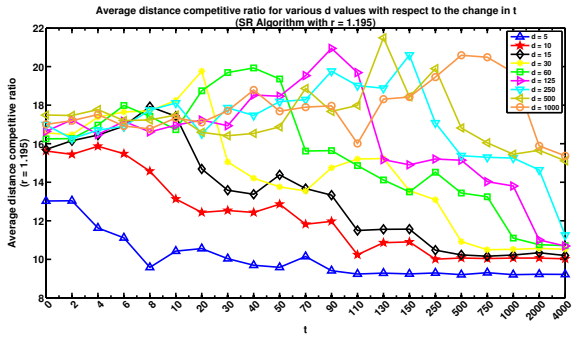


Fig. 4. The average distance competitive performance in simulations of the Algorithm $\mathcal{SR}$ for $r = 1.195$ with respect to the change in d and t.

function of the initial distance and time delay for this particular value of r. Specifically, in the first phase of our simulations, we found the average distance competitive ratios for robot-1 for various r values with respect to changes in distance d and time delay t. Each combination of (r, d, t) was averaged over $5,000$ trials. The simulation results (figure omitted) indicated that $\mathcal{ASR}$ gives the best performance when $1.35 \leq r \leq 1.85$.

In the second phase, we chose the value $r = 1.55$ for more intensive investigation. We ran additional simulations for combinations of (d, t) for this radius. The results are presented in Figure 5, which shows the average competitive ratio for distances $5 \leq d \leq 1000$ and time delays $0 \leq t \leq 1000$. These ratios are bounded above by 22. In fact, with the exception of the combination $d = 500$, $t = 250$, all are less than 20.

The results suggest that the worst-case time delay is near $t = 0$. Indeed, a long delay brings the robot behavior closer to an asymmetric rendezvous search problem (as noted above in the $\mathcal{SR}$ simulations). Again, we see a clear downward trend on the competitive ratio as t increases. For smaller d values, the competitive ratio stabilizes near 6, which is roughly equal the competitive ratio for the SmartCow algorithm value for expansion radius $r = 1.55$.

In summary, our simulations of the $\mathcal{SR}$ and $\mathcal{ASR}$ algorithms show robust performance with respect to time delays. Comparing the simulation plots for $\mathcal{SR}$ and $\mathcal{ASR}$, we see that the $\mathcal{SR}$ algorithm performs better for very small time delays (near-synchronous start), while the $\mathcal{ASR}$ algorithm performs better for very long time delays.

V. Extensions

In this section, we present the two extensions of our algorithm in practical scenarios. First, we consider line segments and half lines (see Figure 6) which can be encountered in large indoor environments. In such environments, robot movements are restricted by the boundary points. We modify our algorithm as follows to utilize these boundaries. If during the execution of $\mathcal{SR}$ the robot hits a boundary, it moves in the opposite direction until it meets with the other robot. Clearly, once a robot encounters the boundary, it knows that the other robot lies in the opposite direction.

Next, we consider simple closed path environments, such as the circular corridor as shown in Figure 7. In the linear case,

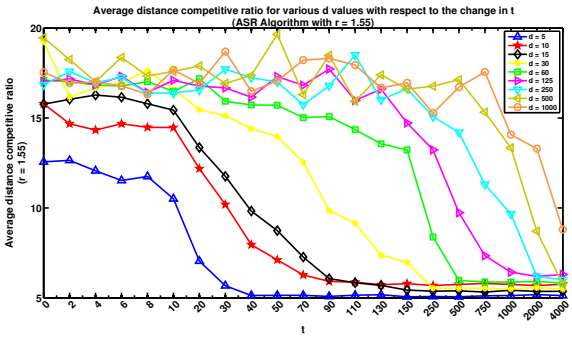


Fig. 5. The average distance competitive performance in simulations of the Algorithm $\mathcal{ASR}$ for $r = 1.55$ with respect to the change in d and t.

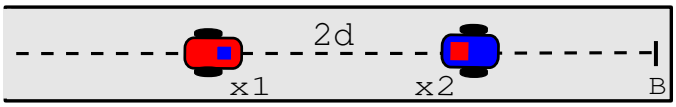


Fig. 6. **The First Extension of $\mathcal{SR}$:** The algorithm $\mathcal{SR}$ can be extended for bounded environments such as line segments and half lines (Lemma 11).

there is only one way for robots to meet: the left robot must move right and the right robot must move left. In a circular environment, eventually there are two ways for the robots to meet: as long as they are not moving in tandem, they are getting closer from one side or the other. In the circular path environment, we simply follow the same $\mathcal{SR}$ algorithm.

It is not difficult to see that the above modifications do not increase the competitive ration of $\mathcal{SR}$. This is summarized in the following lemmas whose proofs are omitted.

Lemma 11: The competitive ratio of the line segment extension does not exceed the competitive ratio of $\mathcal{SR}$.

Lemma 12: The competitive ratio of $\mathcal{SR}$ is an upper bound for the circular path extension.

VI. Experiments

In this section, we present experimental results obtained from a real deployment of two robots. These experiments validate the symmetric rendezvous strategy and its extensions (line segment and circular path) and show its practicability in real world environments.

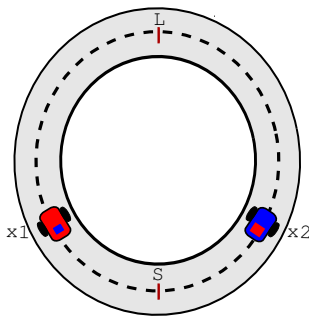


Fig. 7. **Second Extension of $\mathcal{SR}$:** The algorithm $\mathcal{SR}$ can also be extended for closed path like environments (Lemma 12).

Fig. 8. **LEFT:** Experiment setup for the first and second groups showing two iRobot Create robots equipped with Asus EEE PC laptop on it placed on a straight corridor. **RIGHT:** Experiment setup for the third group showing two iRobot Create robots equipped with Asus EEE PC laptop on it placed on a rectangular shaped corridor in an indoor environment.

A. Experimental Setup

We used iRobot Create (see Figure 8) robots as our robotic platform for the experiments. iRobot Create has external sensors such as wheel drop sensors, bumper sensors, cliff sensors and Omnidirectional IR Receiver. In our experiments, we only use bumper sensors for the robot to go to opposite direction when it hits a boundary wall.

We conducted three groups of experiments in indoor environments. We varied the initial distance between the robots in each experiment group. The experiment groups are straight line experiments (SLE), straight line experiments with boundary (SLEB) and closed curve experiments (CCE). In SLE and SLEB, two robots are placed on a straight corridor. In CCE, two robots are placed on a rectangular shaped corridor. The velocity of each robot was set to 0.2 m/s and r is set to 1.195 meters. Robots synchronously execute the $\mathcal{SR}$ Algorithm, or its appropriate variant. If they bump into each other, the experiment ends and we conclude that the rendezvous occurred. We ignore errors that result in robots moving slightly off the line. In all of the experiments, the robots were able to meet successfully. The vertical dashed lines in the experiment figures show the starting points of a new round.

B. Experimental Results

In Figures 9, 10, 11 and 12, we present the robots' trajectories. The experiments in Figures 9, 10 and 11 are performed on the straight corridor and aim to show the practicability of $\mathcal{SR}$ and its first extension. The x-axis in these figures represents the time in seconds and the y-axis represents the x-coordinate of the robots on the line in meters. In Figure 12, we illustrate the results of the experiments performed in a rectangular shaped corridor to show the practicability of the second extension of $\mathcal{SR}$. In this figure, the x-axis and y-axis represent the x-coordinate and y-coordinate of the robots while z-axis represents the time in seconds. When the trajectories intersect, the robots are at the same position on the line and the rendezvous occurs.

Figure 9 represents the experiments in SLE where the robots execute the symmetric rendezvous strategy presented

in Algorithm 1. The robots are initially placed at a distance of three and seven meters. In the top plot of this figure, the robots are placed $2d = 3$ meters away from each other. Solving $d = r^{k+\delta}$ yields $k = 2$ and $\delta = 0.2239$. The rendezvous occurred in two rounds which took 10.3 seconds. The outcome of randomly selected ϵ values by each robot is $\epsilon_1 = 0.22$ and $\epsilon_2 = 0.33$. In the first round Robot-1 tosses heads and moves $r^{2*0+\epsilon_1} = 1.06$ meters to right. Recall that to move synchronously with the other robot, the robot waits long enough to allow the robot to complete its current motion. Therefore in this experiment, the robot waits for $r^{2*0+1} - 1.06 = 0.14$ seconds at the end of the first phase. In the second phase, it moves $r^{2*0+\epsilon_1} + r^{2*0+1+\epsilon_1} = 1.06 + 1.27 = 2.33$ to left and waits for $r^{2*0+1} + r^{2*0+1+1} - 2.33 = 0.31$ seconds at the end of the second phase. Meanwhile, R_2 also tosses heads and moves $r^{2*0+\epsilon_2} = 1.04$ meters to right and then waits for $r^{2*0+1} - 1.04 = 0.16$ at the end of the first phase. In the second phase, it moves $r^{2*0+\epsilon_1} + r^{2*0+1+\epsilon_1} = 1.04 + 1.24 = 2.28$ to left and waits for $r^{2*0+1} + r^{2*0+1+1} - 2.28 = 0.36$ seconds at the end of the second phase.

Figures 10 and 11 represent the experiments in SLEB and SLEB-2b. For both SLEB and SLEB-2b, we place the robots at an initial distance of five and seven meters. In contrast to SLE, in SLEB we consider the case where there is a boundary wall close to either one robot or both robots. As in SLE, the robots execute the bounded segment variant of Algorithm 1. To detect the boundary wall, the robot uses its bumper sensors which are placed at the front side of the robot. Because the sensor must always face the direction of movement, we make the robot turn to face the direction it moves. In SLEB, there is a boundary wall in close vicinity of either Robot-1 or Robot-2. In the top plot of Figure 10, the initial distance between the robots is 5 meters and the boundary wall is located two meters away from Robot-1. In the second plot of Figure 10, the initial distance between the robots is seven meters and the boundary wall is located two meters away from Robot-2. The plots in Figure 11 show the results of the experiments when there are boundary walls in close vicinity of both Robot-1 and Robot-2 (approximately two meters away). In the last plot of Figure 11, Robot-2 tosses tail in the second round and it starts moving to the left. When its displacement from its initial location becomes two meters in the first phase, it hits the boundary wall and starts moving to the right. It keeps moving to the right until it hits Robot-1 at position 6.766. It took 18.2 seconds for the robots to meet.

Figure 12 illustrates CCE. In contrast to the experiments performed on the line, in CCE, both x and y coordinates of the robots change. The robots are placed in a rectangular shaped corridor. The length of the rectangle is ≈ 27.5 and its width is ≈ 18.5 meters. Bottom plot in Figure 8 shows an example to the setup of these experiments. In the top plot of Figure 12, both Robot-1 and Robot-2 start five meters away from the corner. In the bottom plot, Robot-1 starts three meters away from the corner while Robot-2 starts seven meters away from the corner. Each robot executes Algorithm 1, aware that when it reaches the corner of the rectangle, it must turn into to the other corridor and continue its movement. In both plots, the robots met inside one of the corridors.

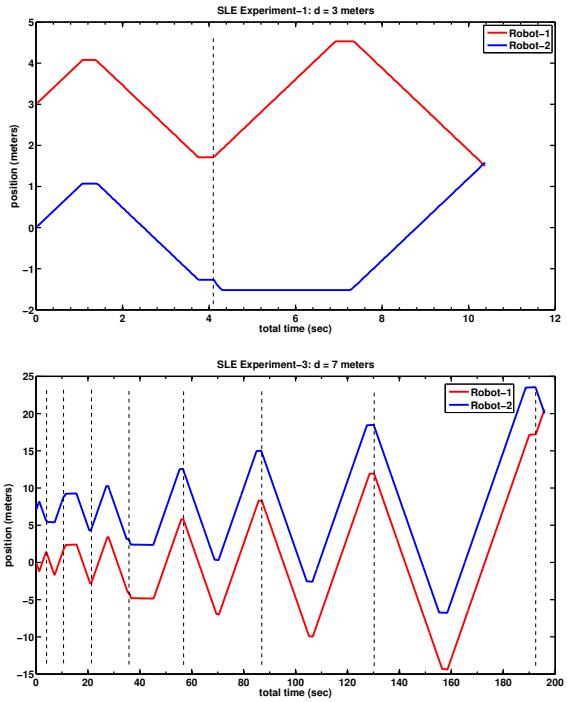


Fig. 9. Straight line experiments. Experiments are performed on a straight corridor. The robots execute Algorithm 1. Rendezvous occurs when the trajectories of the robots intersect. The initial distance between the robots in the top plot is three meters and seven meters in the bottom plot of this figure.

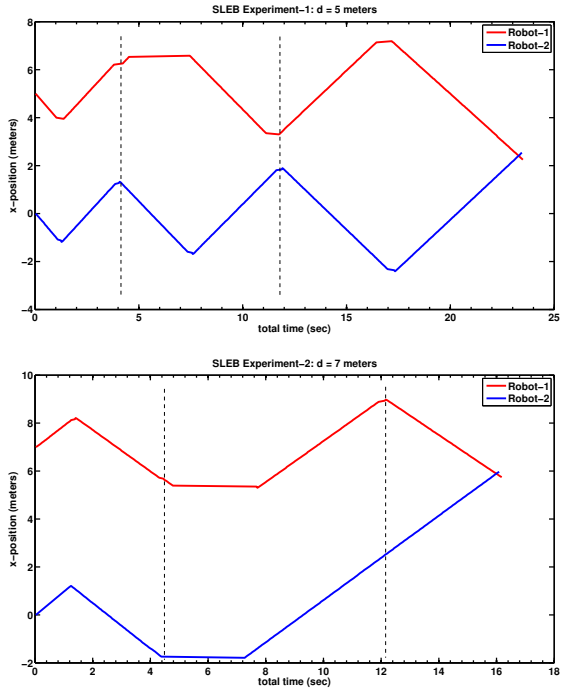


Fig. 10. Straight line experiments with boundary. Experiments are performed on a straight corridor. **SLEB:** There is a boundary wall close to either Robot-1 or Robot-2.

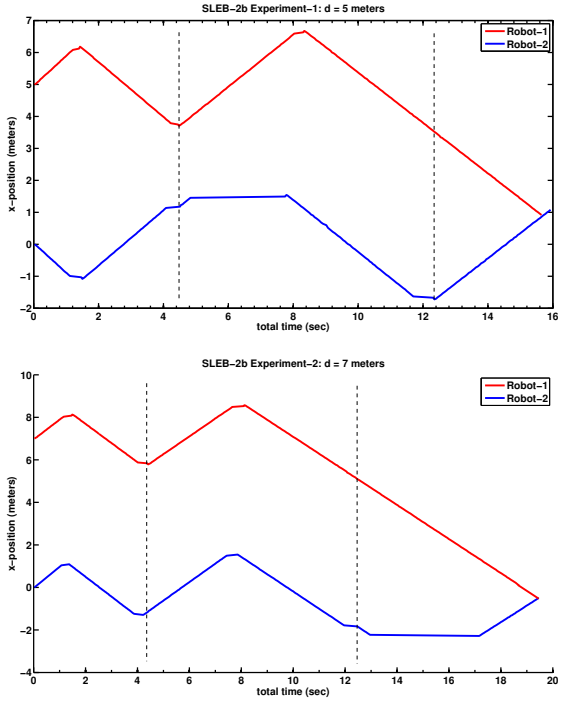


Fig. 11. Straight line experiments with boundary. Experiments are performed on a straight corridor. **SLEB-2b:** There is a boundary wall close to both Robot-1 and Robot-2.

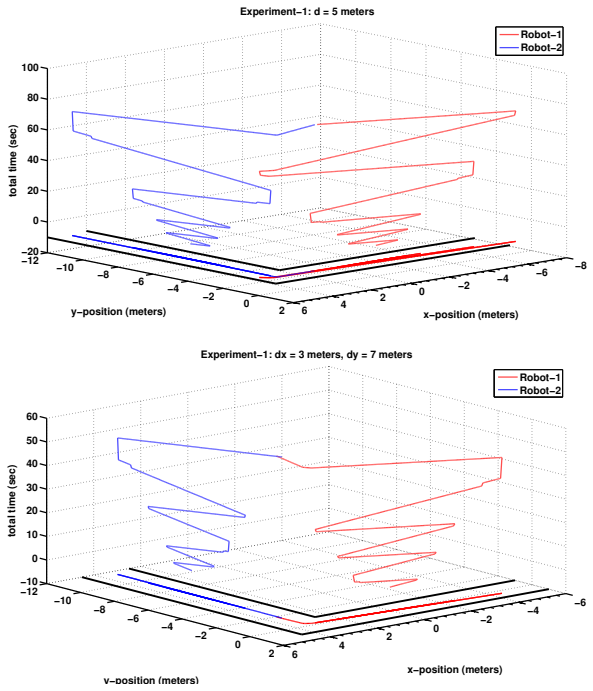


Fig. 12. Closed curve experiments. Experiments are performed on a rectangular shaped corridor. **TOP:** Both Robot-1 and Robot-2 is five meters away from the corner. **BOTTOM:** Robot-1 is three meters away from the corner while Robot-2 is seven meters away from the corner.

VII. Conclusion

In our study of the symmetric rendezvous problem on the line, we generalized the Spiral LostCow algorithm for two symmetric robots. We identified the theoretically optimal value $r = 1.195$. With this expansion radius, our symmetric strategy has a competitive ratio of 17.686 for total distance traveled and a competitive ratio of 24.843 for total time. We showed that our algorithm can be adapted for bounded linear environments and simple closed environments with the same performance guarantees. In simulations, we compared $\mathcal{SR}$'s performance with various r values and various initial distances. We also showed that the algorithm is robust to navigational errors. Our results were consistent with the theoretical analysis. Finally, we report the results from an experiment in which two robots try to meet in a corridor.

In future work, we will study symmetric rendezvous on the line for three or more robots. A further extension is to consider the symmetric rendezvous problem in the plane, both with and without obstacles.

VIII. Acknowledgments

This work is supported in part by National Science Foundation Awards 0916209, 0917676 and 1111638, and a grant from The Scientific and Technological Research Council of Turkey. The authors would like to thank the members of Robotic Sensor Networks Lab at the University of Minnesota for numerous useful discussions.

References

[1] S. Alpern, "The rendezvous search problem," *SIAM Journal on Control and Optimization*, vol. 33, p. 673, 1995.

[2] S. Alpern and S. Gal, *The Theory of Search Games and Rendezvous*. Springer, 2003.

[3] E. Anderson and S. Essegaier, "Rendezvous search on the line with indistinguishable players," *SIAM Journal on Control and Optimization*, vol. 33, p. 1637, 1995.

[4] V. Baston, "Note: Two rendezvous search problems on the line," *Naval Research Logistics*, vol. 46, no. 3, pp. 335–340, 1999.

[5] P. Uthaisombut, "Symmetric rendezvous search on the line using move patterns with different lengths," 2006.

[6] Q. Han, D. Du, J. Vera, and L. Zuluaga, "Improved bounds for the symmetric rendezvous value on the line," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, p. 78, Society for Industrial and Applied Mathematics, 2007.

[7] G. Stachowiak, "Asynchronous Deterministic Rendezvous on the Line," *SOFSEM 2009: Theory and Practice of Computer Science*, pp. 497–508, 2009.

[8] S. Alpern and S. Gal, "Rendezvous search on the line with distinguishable players," *SIAM Journal on Control and Optimization*, vol. 33, p. 1270, 1995.

[9] V. Baston and S. Gal, "Rendezvous on the line when the players' initial distance is given by an unknown probability distribution," *SIAM Journal on Control and Optimization*, vol. 36, p. 1880, 1998.

[10] S. Gal, "Rendezvous search on the line," *Operations Research*, pp. 974–976, 1999.

[11] S. Alpern and A. Beck, "Asymmetric rendezvous on the line is a double linear search problem," *Mathematics of Operations Research*, vol. 24, no. 3, pp. 604–618, 1999.

[12] S. Alpern and A. Beck, "Pure strategy asymmetric rendezvous on the line with an unknown initial distance," *Operations Research*, vol. 48, no. 3, p. 498, 2000.

[13] P. Rawicz, P. Kalata, and K. Murphy, "On the stability of mobile robot rendezvous," in *Intelligent Control (ISIC), 1998. Held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), Intelligent Systems and Semiotics (ISAS), Proceedings*, pp. 570–575, IEEE, 1998.

[14] F. Agah, M. Mehrandezh, R. Fenton, and B. Benhabib, "Rendezvous-guidance based robotic interception," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 3, pp. 2998–3003, IEEE, 2003.

[15] A. Beck and D. J. Newman, "Yet more on the linear search problem," *Israel J. Math.*, vol. 8, pp. 419–229, 1970.

[16] R. Baeza-Yates, J. Culberson, and G. Rawlins, "Searching in the plane," *Information and Computation*, vol. 106, pp. 234–234, 1993.

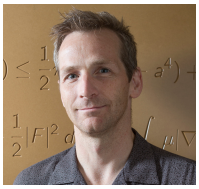
[17] S. Gal, *Search Games*. Academic Press, 1980.

[18] M. Kao, Y. Ma, M. Sipser, and Y. Yin, "Optimal construction of hybrid algorithms," *Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms (SODA '94)*, pp. 372–381, 1994.

[19] A. Beveridge, D. Ozsoyeller, and V. Isler, "Symmetric rendezvous on the line with an unknown initial distance," Tech. Rep. 11-008, University of Minnesota, 2011. `http://www.cs.umn.edu/research/technical_reports.php?page=report&report_id=11-008`.

[20] I. Rekleitis, "A particle filter tutorial for mobile robot localization," Tech. Rep. CIM-04-02, McGill University, 2004.

Deniz Ozsoyeller received her B.S. degree in Computer Engineering from Izmir University of Economics, Turkey, in 2006, ranking first in her department and MSc degree from Ege University, International Computer Institute, Turkey, in 2008. She is currently a teaching assistant in the Computer Engineering Department at Izmir University, Turkey and continues her Ph.D. studies at Ege University, International Computer Institute, Turkey. She received the International Research Fellowship from the Scientific and Technical Research Council of Turkey (TUBITAK). From 2010 to 2012, she was a visiting scholar in the Computer Engineering Department at University of Minnesota under the supervision of Prof. Isler. Her research interests are in robotics and wireless sensor networks.



Andrew Beveridge is an Associate Professor in the Department of Mathematics, Statistics and Computer Science at Macalester College. He previously held a post-doctoral position in the Algorithms, Combinatorics and Optimization group at Carnegie Mellon University. He obtained his MS (1994) and PhD (1997) degrees in Mathematics from Yale University. He obtained his BA (1991) in Mathematics from Williams College. His research interests are primarily in theoretical robotics and probabilistic combinatorics.

Volkan Isler is an Associate Professor in the Computer Science Department at the University of Minnesota. He is currently co-chairing IEEE Society of Robotics and Automation's Technical Committee on Networked Robots. He is also serving as an Associate Editor for IEEE Transactions on Robotics and IEEE Transactions on Automation Science and Engineering. In 2008, he received the National Science Foundation's Young Investigator Award (CAREER). His research interests are primarily in robotics, sensor networks and geometric algorithms.

APPENDIX

In this appendix, we perform some basic calculations required for the analysis in Section III-C. We continue to use the terminology found in that section.

A. Relevant Expectations and Probabilities

Using the substitution $\epsilon = \log_r u$, we have

$$\begin{aligned}\mathbb{E}[r^\epsilon] &= \int_0^1 r^\epsilon d\epsilon = \int_1^r r^{\log_r u}\frac{du}{u\log r} \\ &= \int_1^r \frac{du}{\log r} = \frac{r-1}{\log r}. \end{aligned} \tag{8}$$

Recall that C is the event that $\epsilon_2 \geq 2\delta - \epsilon_1$. We calculate $\mathbb{P}[C]$, $\mathbb{P}[\overline{C}]$, $\mathbb{E}[r^\epsilon \mid C]\mathbb{P}[C]$ and $\mathbb{E}[r^\epsilon \mid \overline{C}]\mathbb{P}[\overline{C}]$. For simplicity, we rename ϵ_1, ϵ_2 as x, y, so that event C becomes $y \geq 2\delta - x$ where $x, y \in [0,1]$.

Finding these values requires integrating in the unit square. Crucially, we note that the limits of integration depend upon δ. Namely, we consider the two cases $0 < \delta \leq 1/2$ and $1/2 < \delta \leq 1$ separately since they require distinct calculations.

1) Case $0 < \delta \leq 1/2$: In this case,

$$\mathbb{P}\left[\overline{C}\right] = \int_0^{2\delta}\int_0^{2\delta - x} \partial y \partial x = 2\delta^2$$

and therefore

$$\mathbb{P}\left[C\right] = 1 - 2\delta^2.$$

We also have

$$\begin{aligned}\mathbb{E}\left[r^{\epsilon_1}|\overline{C}\right]\mathbb{P}[\overline{C}] &= \int_0^{2\delta}\int_0^{2\delta-x} r^x \partial y \partial x \\ &= \frac{r^{2\delta} - 2\delta\log(r) - 1}{\log^2(r)} \end{aligned} \tag{9}$$

and we can evaluate

$$\mathbb{E}\left[r^{\epsilon_1}|C\right]\mathbb{P}[C] = \mathbb{E}[r^\epsilon] - \mathbb{E}\left[r^{\epsilon_1}|\overline{C}\right]\mathbb{P}[\overline{C}].$$

using equations (8) and (9).

2) Case $1/2 < \delta \leq 1$: In this case,

$$\mathbb{P}\left[C \mid \overline{F}\right] = \int_{2\delta-1}^1\int_{2\delta-x}^1 \partial y \partial x = 2(1-\delta)^2$$

and therefore

$$\mathbb{P}\left[\overline{C}\right] = 1 - 2(1-\delta)^2.$$

We also have

$$\begin{aligned}\mathbb{E}\left[r^{\epsilon_1}|C\right]\mathbb{P}\left[C\right] &= \int_{2\delta-1}^1\int_{2\delta-x}^1 r^x \partial y \partial x \\ &= \frac{2(1-\delta)r^2\log(r) - r^2 + r^{2\delta}}{r\log^2(r)} \end{aligned} \tag{10}$$

and we can evaluate

$$\mathbb{E}\left[r^{\epsilon_1}|\overline{C}\right]\mathbb{P}[\overline{C}] = \mathbb{E}[r^{\epsilon_1}] - \mathbb{E}\left[r^{\epsilon_1}|C\right]\mathbb{P}[C]$$

using equations (8) and (10).