Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Symmetric Rendezvous in Planar Environments with and without Obstacles

Deniz Ozsoyeller
Department of Computer Engineering
Izmir University
Gursel Aksel Blvd No:14
Izmir, Turkey 35350
deniz.ozsoyeller@izmir.edu.tr

Volkan Isler
Department of Computer Science and Engineering
University of Minnesota
Keller Hall
200 Union St SE
Minneapolis, MN 55455
isler@cs.umn.edu

Andrew Beveridge
Department of Mathematics, Statistics and Computer Science
Macalester College
1600 Grand Avenue
Saint Paul, MN 55105
abeverid@macalester.edu

Abstract

We study the symmetric rendezvous search problem in which two robots that are unaware of each other's locations try to meet as quickly as possible. In the symmetric version of this problem, the robots are required to execute the same strategy. First, we present a symmetric rendezvous strategy for the robots that are initially placed on the open plane and analyze its competitive performance. We show that the competitive complexity of our strategy is $O(d/R)$ where d is the initial distance between the robots and R is the communication radius. Second, we extend the symmetric rendezvous strategy for the open plane to unknown environments with polygonal obstacles. The extended strategy guarantees a complete coverage of the environment. We analyze the strategy for square, translating robots and show that the competitive ratio of the extended strategy is $O(d/D)$ where D is the length of the sides of the robots. In obtaining this result, we also obtain an upper bound on covering arbitrary polygonal environments which may be of independent interest.

Introduction

How quickly can two robots who do not know each others' locations meet? This problem is known as rendezvous search. It has applications in scenarios where two robots dropped off from a plane may be trying to get together after deployment. Similarly, robots may need to find each other after completion of independent tasks.

There are two primary versions of the rendezvous search problem depending on whether or not the players can execute different strategies. In asymmetric rendezvous search, the players can choose separate roles in advance and execute distinct strategies. For example, one can wait for the other. In the second version, called symmetric rendezvous search, the players must execute the same strategy. Symmetric rendezvous strategies are appealing for robotics applications as they eliminate the need to implement and maintain a different program for each robot.

In this paper, we study the symmetric rendezvous search problem in planar environments. We start with rendezvous on the open plane. This version is readily applicable for robots operating on large open fields or water surface. It can also be used by robots operating at a constant height (resp. depth) in the air (resp. underwater). In the second part of the paper, we show how the strategy can be extended to environments with (unknown) obstacles.

We prove the performance of our algorithms using competitive analysis. In a given environment, if the robots know each others' locations, their optimal strategy would be to move toward each other along the shortest path between them. In competitive analysis, one seeks for the worst case deviation of an online strategy from this optimal offline behavior over all possible initial locations in all possible environments. This worst case ratio is known as the competitive ratio.

First, we present a randomized strategy for symmetric rendezvous on the open plane. While the strategy is relatively simple, its competitive analysis is rather involved. Assuming that the robots move at unit speed, we show that the competitive ratio of our strategy for rendezvous in the open plane is $O(d/R)$ where d is the initial distance between the robots and R is their communication radius.

Next, we study environments with polygonal obstacles and design a rendezvous strategy for square robots of edge length D translating in the environment. The strategy is an extension of the first strategy in which each robot independently performs a modified depth first search on a grid with cell size D. We prove a competitive ratio of $O(d/D)$ (each robot maintains a separate grid). In obtaining this result, we also provide a strategy for covering an unknown polygonal environment which may be of independent interest. In the next section, we start with an overview of related work.

Related Work

Various versions of the rendezvous problem on the plane have been studied. Asymmetric version of the rendezvous problem in a two dimensional region is studied in (Thomas and Hulme 1997; Anderson and Fekete 2001; Roy and Dudek 2001). Asynchronous rendezvous in two dimensional region is studied in (Flocchini et al. 2001; Souissi, Défago, and Yamashita 2006; Ganguli, Cortes, and Bullo 2006; Lin,

Morse, and Anderson 2007; Fang, Morse, and Cao 2008; Jurek, Leszek, and Pelc 2009; Czyzowicz, Labourel, and Pelc 2010; Czyzowicz et al. 2010; Collins et al. 2010). (Thomas and Hulme 1997) simulate the problem of a rescue helicopter searching for a lost walker in which the speed and radius detection of the players are not identical. (Anderson and Fekete 2001) consider the case where players know that they are at a distance d apart, but they do not know the direction in which they should travel. They also consider a situation in which *player-1* knows the initial position of *player-2*, while *player-2* is only given information on the initial distance of *player-1*. (Roy and Dudek 2001) consider the problem of rendezvous between two robots which can select potential rendezvous locations, called *landmarks* as they explore an unknown environment.

The symmetric version of the rendezvous problem in a two dimensional region is studied in (Ruckle 2007). They consider the rendezvous problem with *s* seekers in a rectangular lattice of locations. Each player knows the distribution of the seekers at time zero and its own location, but not the location of any other. The main result for this paper is that for any dimension of lattice and for any initial distribution of seekers, there are optimal strategies for the seekers that converge. (Alpern and Street 2001) study both the asymmetric and symmetric versions of the rendezvous problem with two players on the *n* dimensional integer lattice: Two players who have no common notion of locations or directions. They are initially placed at nodes whose difference vector has length 2 and is parallel to some coordinate axis. In each period they must move to an adjacent node. In our present work, we present the first results for symmetric rendezvous with unknown initial location on the plane and in environments with obstacles.

Our work is also related to the work of (Gabriely and Rimon 2005) who present an online motion planning algorithm for a mobile robot to find a stationary target in an unknown environment. Similar to our formulation, their results are for a robot of size $D \times D$. The robot covers disks of increasing size in each round to find the target. In their model, they assume that the coverage cost of a free cell and a partially occupied cell are the same. A subroutine of our algorithm extends this strategy and explicitly computes an upper bound on the cost of covering a partially occupied cell. Other work on finding stationary targets include search in environments with m radial corridors (Baezayates, Culberson, and Rawlins 1993) and rectangular environments (Berman et al. 1996).

Symmetric Rendezvous Search in Planar Environments

In this section, we study the symmetric rendezvous search problem with two robots in the plane that is not populated by obstacles. The robots do not know each other's locations or their global positions in the plane. They further do not know the initial distance between each other. Let the expansion radius $r > 1$ be fixed (we determine the optimal choice for r in the next subsection). The distance between the robots is $d = r^{k+\delta}$ where $k \in +\mathbb{Z}$ and $\delta \in (0, 1]$. We assume that there is a non-zero communication range R such that the rendezvous is established if the distance between robots is less than R.

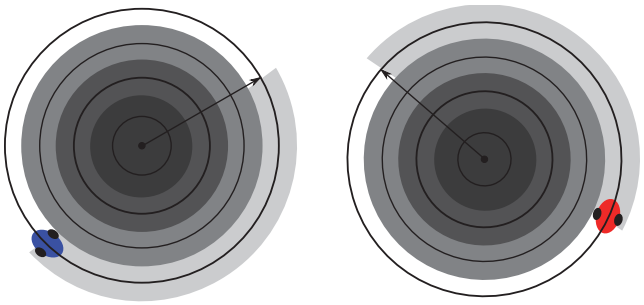

Figure 1: Overview of the strategy: Each robot flips a coin to decide whether to move or wait. If heads, the robot follows concentric circles centered at the robot's initial location. If tails, it waits for the distance it must move in current round i While drawing a circle, the robot sweeps a ring of width $2R$ where R is the communication range of the robot.

Algorithm 1 shows our symmetric synchronous rendezvous strategy on the plane. We divide the strategy into rounds. At the beginning of each round, each robot flips a coin to determine whether to move or to wait. Algorithm- 2 shows how the robots move if the outcome of the coin toss is head in round i. A robot covers a disk of radius $r^i + R$ in round i by drawing concentric circles (inward to outward) centered at the robot's initial location. The radius of the outermost circle is $r^i + R$. While drawing a circle, the robot sweeps a ring of width $2R$. Therefore, we increase the radius of the circle a robot draws by $2R$ until it reaches to the outermost circle. The robot follows the motion pattern given in Algorithm 2. The robot first executes line 3 and moves R units to the north and draws a disk of radius R. Then it moves $2R$ units to the north and draws a disk of radius $3R$. It follows the same motion pattern until it reaches to radius $(2N + 1)R$ units to north from its initial location in round i. Then, it moves $(2N+1)R$ units south to come back to its initial location. Figure 1 shows the moving pattern of a robot executing the algorithm. If the outcome is tail, the robot waits for the distance it must move in current round i to be in synch with the other robot (see Algorithm- 1, Line 11). We next prove the competitive ratio of our algorithm.

The Analysis

In this section, we analyze our algorithm's performance and find an upper bound on the expected distance traveled by the robot. We only show the performance of robot-1 in our analysis. Due to the symmetric strategies, the performance of the robot-2 is the same. In order to obtain bounds on the performance of Algorithm 1 (distance competitive-ratio), we divide its execution into two stages: round $i \leq k$ and round $i \geq (k + 1)$. We give an overview of each stage before diving into the analysis. Stage-1 is an initialization

Algorithm 1 $\mathcal{SP}_1$: Symmetric synchronous rendezvous strategy on the plane

```
r ← 1.225
R ← 1
i ← 1
head ← 0
tail ← 1
waitTime ← 0
outcome ← random from {0,1}
while checkRendezvous() != true do
    N ← ⌊r^i / 2R⌋
    if outcome = tail then
        waitTime ← (2N + 1)R + 2π(N + 1)^2 R
        wait(waitTime)
    else if outcome = head then
        move(R, N)
    end if
    i ← i + 1
    outcome ← random from {0,1}
end while
```

Algorithm 2 $Procedure_1$: move(R, N)

```
for j = 0 to N do
    if j = 0 then
        move R units to north
    else
        move 2R units to north
    end if
    move in a circle that has a radius (2j + 1)R
end for
move (2N + 1)R to south
```

stage whose length depends on the (unknown) initial distance $d = r^{k+\delta}$. This stage encompasses the early rounds in which the robots do not move far enough to meet. In Stage-2, the rendezvous behavior is consistent, so we can compute the expected travel distance using an infinite sum.

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: A_i is the event that robot-1 is moving in round i; B_i is the event that robot-2 is moving in round i. The event S_i will be our proxy event for success. For $i \leq k$, we have $\mathbb{P}[S_i] = 0$ because the robots do not travel far enough to meet. For $i > k$, we define $S_i = (A_i \wedge \overline{B}_i) \vee (\overline{A}_i \wedge B_i)$. Note that this definition purposely omits the event that the robots meet while both are on the move, but the probability of this event is zero, unless the initial robot configuration obeys an exceptional geometry. 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 $i > 0$. Furthermore, we have

$$\mathbb{P}[R_i] = \begin{cases} 1 & 0 \leq i \leq k, \\ 2^{-i+k} & k < i. \end{cases} \tag{1}$$

To find an upper bound on the expected distance traveled by the robot, we consider that the rendezvous only occurs when one robot is moving while the other robot is waiting and they travel far enough to meet. The algorithm terminates when the robots meet.

We are now ready to study Algorithm 1. Next two sections present the analysis of stages 1 and 2 respectively. We prove the following bound on the algorithm's distance competitive ratio, and its time competitive ratio.

Theorem 1. *The best choice for the expansion radius of Algorithm 1 is* $r = \sqrt{1.5} \approx 1.225$. *This gives a distance competitive ratio of*

$$7.854\frac{d}{R} + 40.8577\frac{R\log d}{d} + 16.0664\frac{R}{d} + 55.3236$$

The time competitive ratio is twice this bound.

Here, $\log d$ denotes the natural logarithm.

Analysis of Stage-1

We compute the expected distance traveled during round $i \leq k$. The robots can not meet in this stage since their displacement is at most $r^k < r^{k+\delta}$. Suppose that event A_i holds: the robot is active during round i. Let

$$N = \left\lfloor \frac{r^i}{2R} \right\rfloor \leq \frac{r^i}{2R}.$$

Over the course of the round, the robot travels $(2N+1)R \leq r^i + R$ units north before covering this distance again to return to its initial location. The robot traverses the circumference of each circle of radius $(2j+1)R$ for $0 \leq j \leq N$. The expected distance traveled in this stage is

$$\begin{aligned} \mathbb{E}[D_i \mid A_i] &\leq 2\left(r^i + R\right) + 2\pi R \sum_{j=0}^{N} (2j+1) \\ &= 2\left(r^i + R\right) + 2\pi R(N+1)^2 \\ &\leq 2\left(r^i + R\right) + \frac{\pi\left(r^i + 2R\right)^2}{2R} \\ &= \frac{\pi}{2R} r^{2i} + 2(1+\pi)(r^i + R). \end{aligned} \tag{2}$$

We now compute the expected distance traveled in this stage.

Lemma 2. *The expected distance traveled during Stage-1 is bounded by*

$$\sum_{i=1}^{k} \mathbb{E}\left[D_i\right] < d^2\left(\frac{\pi}{4R}\right)\frac{r^2}{r^2-1} + (1+\pi)\left(d \cdot \frac{r}{r-1} + (1+\log_r d)R\right).$$

Proof. Since the robots to do not travel far enough to meet, every round up to k is performed. In other words, $\mathbb{P}[R_i]$=1 for $0 \leq i \leq k$. Within a given round, the robot is active half

of the time. Using equation (2), we find that the expected distance traveled satisfies

$$\sum_{i=0}^{k} \mathbb{E}\left[D_i\right] = \sum_{i=0}^{k} \mathbb{E}[D_i|R_i]\mathbb{P}[R_i] = \frac{1}{2}\sum_{i=0}^{k} \mathbb{E}[D_i|A_i]$$
$$< \frac{\pi}{4R}\sum_{i=0}^{k} r^{2i} + (1+\pi)\left(\sum_{i=0}^{k} r^i + (k+1)R\right)$$
$$= \frac{\pi}{4R}\cdot\frac{r^{2k+2}-1}{r^2-1} + (1+\pi)\left(\frac{r^{k+1}-1}{r-1} + (k+1)R\right)$$
$$< d^2\left(\frac{\pi}{4R}\right)\frac{r^{2-2\delta}}{r^2-1} + (1+\pi)\left(d\frac{r^{1-\delta}}{r-1} + (1+\log_r d)R\right),$$

which is maximized for $\delta = 0$. □

Analysis of Stage-2

In this section, we consider rounds $i \geq (k+1)$. We separately calculate the expected distance traveled during unsuccessful rounds and the expected distance traveled in the (final) successful round.

Lemma 3. *The expected total distance traveled during unsuccessful Stage-2 rounds is bounded by*

$$\sum_{i=k+1}^{\infty} \mathbb{E}\left[D_i \mid \overline{S}_i\right] < d^2\frac{\pi r^2}{8R(2-r^2)} + \frac{(1+\pi)}{2}\left(d\cdot\frac{r}{2-r} + R\right).$$

Proof. We bound the expected distance traveled by robot-1; the calculation for robot-2 is analogous. Assuming that the algorithm is active in round i, we know that the robots rendezvous precisely when $S_i = (A_i \wedge \overline{B}_i) \vee (\overline{A}_i \wedge B_i)$ occurs. Furthermore, the distance traveled by robot-1 is nonzero only when A_i holds. This means that $\mathbb{E}[D_i \mid \overline{S}_i \wedge R_i] = \mathbb{E}[D_i \mid A_i \wedge B_i \wedge R_i]$, and $\mathbb{E}[D_i \mid S_i \wedge R_i] = \mathbb{E}[D_i \mid A_i \wedge \overline{B}_i \wedge R_i]$.

Using equations (1) and (2), we find that the total expected distance traveled in unsuccessful Stage-2 rounds is

$$\sum_{i=k+1}^{\infty} \mathbb{E}[D_i \mid A_i \wedge B_i \wedge R_i]\mathbb{P}[A_i \wedge B_i \wedge R_i]$$
$$< \sum_{i=k+1}^{\infty}\left(\frac{\pi}{2R}r^{2i} + 2(1+\pi)(r^i+R)\right)\left(\frac{1}{2}\right)^{i-k+2}$$
$$= \sum_{j=0}^{\infty}\left(\frac{\pi}{2R}r^{2(j+k+1)} + 2(1+\pi)(r^{j+k+1}+R)\right)\left(\frac{1}{2}\right)^{j+3}$$
$$= \frac{\pi r^{2k+2}}{16R}\sum_{j=0}^{\infty}\left(\frac{r^2}{2}\right)^j + \frac{(1+\pi)}{4}\left(r^{k+1}\sum_{j=0}^{\infty}\left(\frac{r}{2}\right)^j + 2R\right)$$
$$= \frac{\pi r^{2k+2}}{8R(2-r^2)} + \frac{(1+\pi)}{2}\left(\frac{r^{k+1}}{2-r} + R\right)$$
$$= d^2\frac{\pi r^{2-2\delta}}{8R(2-r^2)} + \frac{(1+\pi)}{2}\left(d\cdot\frac{r^{1-\delta}}{2-r} + R\right),$$

which is maximized for $\delta = 0$. □

Lemma 4. *The expected distance traveled during the successful round is bounded by*

$$\sum_{j=k+1}^{\infty} E[D_j \mid S_j] \leq \frac{\pi}{8R}d^2 + \frac{1+2\pi}{4}(d+R).$$

Proof. We have $S_j = (A_j \wedge \overline{B}_j) \vee (\overline{A}_j \wedge B_i)$, and only the event $A_j \wedge \overline{B}_j$ contributes to the distance traveled by robot-1. In a successful round, robot-1 explores a circular area with radius $r^{k+\delta}$ before discovering robot-2. The total distance traveled by robot-1 is calculated similarly to equation (2), using $r^i = r^{k+\delta} = d$. Independent of j, we have

$$E[D_j \mid A_j \wedge \overline{B}_j \wedge R_j] \leq (d+R) + \frac{\pi(d+2R)^2}{2R}$$
$$= \frac{\pi}{2R}d^2 + (1+2\pi)(d+R).$$

Therefore, the expected distance traveled in successful rounds during Stage-2 is

$$\sum_{j=k+1}^{\infty} \mathbb{E}[D_j \mid A_j \wedge \overline{B}_j \wedge R_j]\mathbb{P}[A_i \wedge \overline{B}_i \wedge R_i]$$
$$= \left(\frac{\pi}{2R}d^2 + (1+2\pi)(d+R)\right)\sum_{i=k+1}^{\infty}\left(\frac{1}{2}\right)^{i-k+2}$$
$$= \frac{\pi}{8R}d^2 + \frac{1+2\pi}{4}(d+R).$$

□

Computing the Competitive Ratio

Having found bounds for the expected distance traveled in both stages, we are ready to calculate the distance competitive ratio of our algorithm.

Proof of Theorem 1. The expected distance traveled is the sum of the bounds in Lemmas 2, 3 and 4. To obtain competitive ratio, we divide this sum by $d/2$, which yields

$$\frac{d\pi}{2R}\left(\frac{r^2}{r^2-1} + \frac{r^2}{2\left(2-r^2\right)} + \frac{1}{2}\right) +$$
$$\left(\frac{2\left(1+\pi\right)r}{r-1} + \frac{(1+\pi)\,r}{(2-r)} + \frac{1}{2} + \pi\right) +$$
$$\frac{R}{d}\left(2(1+\pi)\log_r d + 4\pi + \frac{7}{2}\right)$$

The $\Theta(d)$ dominates this expression for large d, so we choose our r value to minimize the coefficient of d. The best value is $r = \sqrt{1.5} \approx 1.225$, giving the distance competitive ratio specified in the theorem.

The analysis for the time competitive ratio is analogous to the distance calculations. In each round, a robot is inactive with probability $1/2$. In this case, the robot wait time in round i is identical to the time for an active robot to complete the round (since the robots move unit distance in unit

time). Therefore, doubling the bounds in each of the three lemmas gives a bound on the expected time until completion. Of course, optimal $r = \sqrt{1.5} \approx 1.225$ is same for the time competitive ratio. □

Symmetric Rendezvous in Environments with Obstacles

In this section, we study the symmetric rendezvous problem in the environments with obstacles. In such environments, the strategy presented in the previous section which is designed for the obstacle-free environments is not applicable. This is because the robot cannot cover the entire environment by moving in a circular fashion due to the obstacles. However, the strategy can be extended to solve the same problem for environments with obstacles. As in the obstacle-free case, the initial distance between the robots (length of the shortest path) is given as $d = r^{k+\delta}$ where $\delta \in (0, 1]$ and $k \in \mathbb{Z}^+$. To formulate the problem, we make the following assumptions.

1. The environment is unknown.
2. The obstacles in the environment are polygonal.
3. The robot is a translating robot (it does not rotate).
4. The robots are square robots of size D by D and they are equipped with obstacle detection sensors.

We present the algorithm in the configuration-space (C) in which the square robot is represented as a point robot. We construct C-$obstacles$ by taking the Minkowski sum of every obstacle with D. For technical reasons, we treat C-$obstacle$ as open sets without the boundary.

Symmetric Rendezvous Search Algorithm

Similar to Algorithm SR, the algorithm proceeds in rounds indexed by $i \in \mathbb{Z}^+$. At the beginning of each round, the robot flips a coin to decide whether to move or wait in that round. Recall that we represent the robots as points in their configuration space C. The randomized decision of the robots is as follows: *If the robot tosses heads*, it draws a disk of radius r^i centered at its initial location where r is the expansion radius of the robot (optimized in Theorem 7). Let $Disk_i$ denote the disk with radius r^i in round i. The robot first discretizes the continuous area in $Disk_i$ to a grid of D size cells. Then it covers the grid cells in $Disk_i$ by the coverage algorithm (explained in the next subsection). *If the robot tosses tails*, it waits for the maximum time it takes for the robot to cover $Disk_i$. (In the next section, we present a bound on the coverage time of $Disk_i$.) The rendezvous occurs when the robots are in the same cell and one robot is moving while the other robot is waiting. The pseudocode of the rendezvous strategy is presented in Algorithm 3 and waitCoverage procedure is presented in Algorithm 4. We next explain the coverage algorithm.

Algorithm 3 SR-$obstacle$: Symmetric rendezvous search algorithm.

Input: C-obstacle with removed boundary of the expanded obstacles.

```
r ← 1.123
i ← 0
coinFlip ← random from {−1, 1}
while checkRendezvous() != true do
    if coinFlip = -1 then
        Draw a disk with a radius r^i
        C_current ← cell the robot is initially located.
        createGrid(C_current, r^i)
        CA(Disk_i)    //cover disk
    else if coinFlip = 1 then
        waitCoverage(i)
    end if
    i ← i + 1
    coinFlip ← random from {−1, 1}
end while
```

Coverage Algorithm

The robots execute the coverage algorithm (CA) in grid cells in $Disk_i$. The grid cells can be free of obstacles or full/partially occupied by the obstacles (see Figure 2). CA is basically a modified version of DFS algorithm and guarantees that every point in the environment is covered. We now present the details of CA.

We define the following variables used in our algorithm: C_{center} is the cell the robot is initially located in the grid; $C_{current}$ is the cell the robot is currently moving in; C_{parent} is the cell from which the robot enters $C_{current}$; C^j_{child} is one of the neighbor cells of $C_{current}$. Two cells are neighbors if they share a common edge. The shared edge between the neighbor cells can be intersected by obstacles. For example, in Figure 2, neighbors of Cell-A are Cells (B, E, F and H). Initially $C_{current}$ is set to C_{center}.

The robot covers each grid cell in the DFS tree by following the obstacle/cell boundary in a counter clockwise fashion. We refer to this motion plan as "CCWBF" (counter clockwise boundary following) in the rest of the paper. Note that there can be one entrance of a cell while there can be many exit points of a cell. We define e_j as the entrance of C^j_{child} (which is equal to the exit of $C_{current}$). If the robot in $C_{current}$ hits the boundary of C^j_{child}, C_{parent} becomes $C_{current}$ and C^j_{child} becomes the new $C_{current}$. The robot then starts covering new $C_{current}$ by CCWBF until it hits the boundary of C^j_{child}. It then recursively covers C^j_{child} and all its children and continues CCWFB until it reaches e_j. When the robot comes back to e_j, we say that the cell is covered. The robot then enters C_{parent} from $exit$ of C_{parent}. At this point, C_{parent} becomes $C_{current}$ and the robot continues covering $C_{current}$ by CCWBF again until it reaches to the entrance of $C_{current}$. All the other cells in the grid are covered in a similar fashion.

Figure 2 illustrates an example execution of the coverage algorithm on a 3×3 grid. The cells are indexed between A-I and the robot is initially located at Cell-A. Arrows show the direction the robot moves in a cell and the stars show $entrance$ of each cell. The resulting DFS tree which represents the order the robot visits the grid cells is shown in

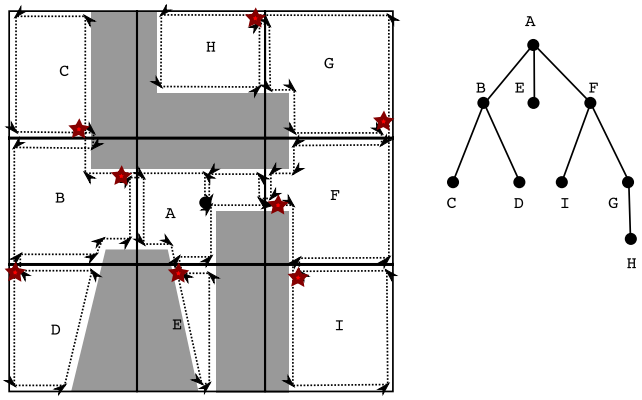

Figure 2: **(a)** A sample run of the coverage algorithm on a 3×3 grid. The grid cells are indexed A-I. **(b)** DFS Tree.

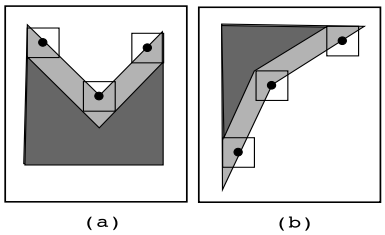

Figure 3: Concave corner of obstacle P and its corresponding corner in C-*obstacle*. (a) If E_i and E_o of v are not in opposite quadrants. (b) If E_i and E_o of v are in opposite quadrants.

Figure 2 (b). The robot first moves in upward direction in Cell-A to start CCWBW. When it hits the boundary of Cell-B, it follows CCWBW in this cell until it hits the boundary of cell-C. Since cell-C has no children, the robot comes back to $entrance$ where it finishes covering Cell-C. The robot then continues CCWBF in Cell-C until it hits the boundary of Cell-D. All the other grid cells are covered in a similar way by the robot.

Procedure 4 waitCoverage(i)

Input: D
1: $N_i \leftarrow \pi r^{2i}/D^2$
2: wait($8N_iD$)

The Analysis

Due to space limitations we provide a brief sketch of the analysis.

We first study interactions between C-obstacles and grid cells.

Lemma 5. *A corner of a C-obstacle is always an obtuse angle.*

Proof. Let E_i denote the incoming edge and E_o denote the outgoing edge of the corner of an obstacle P. A corner of P can be either convex (has an interior angle ≤ 180 degrees) or concave (has an interior angle > 180 degrees). Let v be a concave corner of P. We first examine how v is mapped to a C-$Obstacle$ corner. If E_i and E_o of v are not in opposite quadrants, the robot sweeps E_i and E_o with its two different corners. Figure 3 (a) shows an example for this case. If E_i and E_o of v are in opposite quadrants, then the robot sweeps E_i and E_o with its same corner. An example for this case is illustrated in Figure 3 (b).In both cases, v remains as a concave corner in C-*obstacle*. Let u be a convex corner of P. Second, we examine how u is mapped to a C-$Obstacle$ corner. We divide the second case into three subcases according to E_i and E_o being in the same or different quadrants.

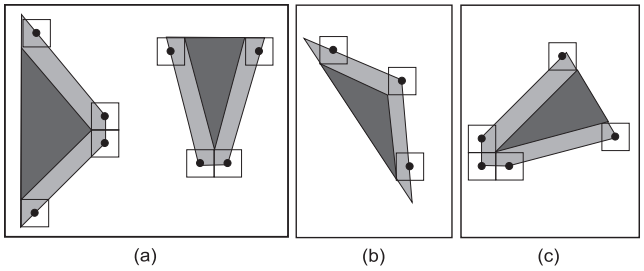

Figure 4: Convex corner of obstacle P and its corresponding corner in C-*obstacle*. (a) If E_i and E_o are in different but not in opposite quadrants. (b) If E_i and E_o are in opposite quadrants. (c) If E_i and E_o are in the same quadrant.

1. **Subcase-1 (different but not in opposite quadrants):** If E_i and E_o are in different but not in opposite quadrants, then Minkowski sum introduces a new edge in the C-*obstacle*. This edge is either vertical/horizontal and has a length D. Therefore in this case, u corresponds to two obtuse angle (> 90) corners in C-*obstacle*. Figure 4 (a) illustrates two examples for this case.
2. **Subcase-2 (opposite quadrants):** If E_i and E_o are in opposite quadrants (meaning that u is with an obtuse angle), then the robot sweeps E_i and E_o with its same corner. Hence, the corresponding C-*obstacle* corner of v still remains as an obtuse angle. Figure 4 (b) shows an example for this case.
3. **Subcase-3 (same quadrants):**If E_i and E_o are in the same quadrant, Minkowski sum introduces two new horizontal and vertical edges in the C-*obstacle*. This yields an obtuse angle followed by an orthogonal corner which is followed by another obtuse angle corner. Horizontal and vertical edges have length D. Figure 4 (c) shows an example for the subcase where E_i and E_o are in the same quadrant.

□

Lemma 5, along with the observation that a C-*obstacle* has edge length at least D, yields an interesting structural property: C-*obstacles* can divide a cell into at most two regions. The proof, illustrated in Figure 5, shows that the presence of a third C-obstacle contradicts Lemma 5.

Next, we prove the following lemma:

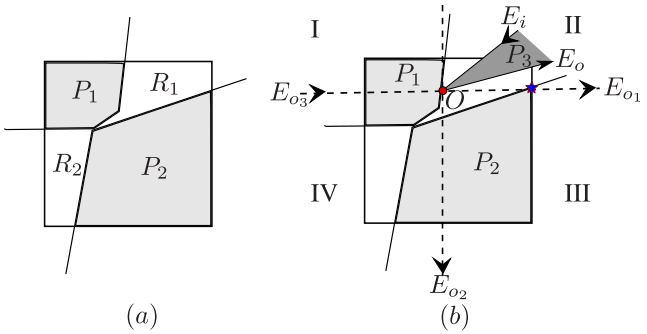

Figure 5: (a) P_1 and P_2 are two C-*obstacles* in a cell. They split the cell into two (light shaded) regions. (b) We now introduce a third obstacle, P_3 in order to create a third (dark shaded) region in the cell. R_1 and R_2 denote the free regions in the cell. E_i and E_o denote the incoming and outgoing edges to P_3. An imaginary quadrant is drawn and the corner of P_3 is fixed to its origin O. Regions of the quadrant are indexed clockwise between I-IV. E_{o_1}, E_{o_2} and E_{o_3} denote the imaginary outgoing edges of P_3.

Lemma 6. *The cost (in terms of distance traveled) of covering a region is* $T_c < 4D$.

Since each grid cell is composed of at most two regions, we obtain an upper bound on covering a grid cell: $T_c \leq 8D$.

This allows us to bound the total time cover $Disk_i$: $T_{D_i} < 8N_iD$. Here, N_i denotes the number cells inside $Disk_i$.

The remaining analysis of Algorithm SR-*obstacle* is similar to the obstacle-free case. The only difference is the distance traveled by the robot in CA which is upper bounded by $8N_iD$. This yields our main result:

Theorem 7. *The optimal expansion radius* r *for the symmetric rendezvous algorithm for the environments with obstacles is* $r = 1.123$. *This choice of* r *gives an algorithm that has a competitive ratio of* $292.968\frac{d}{D}$.

Concluding Remarks

We studied the symmetric rendezvous problem in planar settings and presented randomized algorithms for two robots to meet. We presented the details of the competitive analysis of the first algorithm designed for the open plane. We then extended this algorithm for environments with polygonal obstacles and presented an overview of its competitive analysis.

Our future work includes implementing these strategies on real robots. We will extend our results to handle asynchronous robots that start their search at different times. We will also investigate algorithms for symmetric multi-robot rendezvous.

Acknowledgments

Volkan Isler is supported in part by National Science Foundation Awards 1111638, 0916209, 0917676, and 0936710. Deniz Ozsoyeller is supported by a grant from The Scientific and Technological Research Council of Turkey.

References

Alpern, S., and Street, H. 2001. Rendezvous in One and More Dimensions. *CDAM Research Report LSE-CDAM-2001-05*.

Anderson, E., and Fekete, S. 2001. Two dimensional rendezvous search. *Operations Research* 107–118.

Baezayates, R.; Culberson, J.; and Rawlins, G. 1993. Searching in the plane. *Information and Computation* 106(2):234–252.

Berman, P.; Blum, A.; Fiat, A.; Karloff, H.; Rosen, A.; and Saks, M. 1996. Randomized robot navigation algorithms. In *Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms*, 75–84. Society for Industrial and Applied Mathematics.

Collins, A.; Czyzowicz, J.; Gasieniec, L.; and Labourel, A. 2010. Tell me where I am so I can meet you sooner. In *icalp*.

Czyzowicz, J.; Ilcinkas, D.; Labourel, A.; and Pelc, A. 2010. Asynchronous deterministic rendezvous in bounded terrains. *Structural Information and Communication Complexity* 72–85.

Czyzowicz, J.; Labourel, A.; and Pelc, A. 2010. How to meet asynchronously (almost) everywhere. *Arxiv preprint arXiv:1001.0890*.

Fang, J.; Morse, A.; and Cao, M. 2008. Multi-agent rendezvousing with a finite set of candidate rendezvous points. In *American Control Conference, 2008*, 765–770.

Flocchini, P.; Prencipe, G.; Santoro, N.; and Widmayer, P. 2001. Gathering of asynchronous oblivious robots with limited visibility. *STACS 2001* 247–258.

Gabriely, Y., and Rimon, E. 2005. Competitive complexity of mobile robot on line motion planning problems. *Algorithmic Foundations of Robotics VI* 155–170.

Ganguli, A.; Cortes, J.; and Bullo, F. 2006. Multirobot rendezvous with visibility sensors in nonconvex environments. *Arxiv preprint cs/0611022*.

Jurek, C.; Leszek, G.; and Pelc, A. 2009. Gathering few fat mobile robots in the plane. *Theoretical Computer Science* 410(6-7):481–499.

Lin, J.; Morse, A.; and Anderson, B. 2007. The multi-agent rendezvous problem. Part 2: The asynchronous case. *SIAM Journal on Control and Optimization* 46(6):2120–2147.

Roy, N., and Dudek, G. 2001. Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations. *Autonomous Robots* 11(2):117–136.

Ruckle, W. 2007. Rendez-vous search on a rectangular lattice. *Naval Research Logistics* 54(5):492–496.

Souissi, S.; Défago, X.; and Yamashita, M. 2006. Gathering asynchronous mobile robots with inaccurate compasses. *Principles of Distributed Systems* 333–349.

Thomas, L., and Hulme, P. 1997. Searching for targets who want to be found. *Journal of the Operational Research Society* 48(1):44–50.