Symmetry Breaking in the Plane: Rendezvous by Robots with Unknown Attributes

Jurek Czyzowicz Leszek Gąsieniec Ryan Killick Evangelos Kranakis

Abstract

We study a fundamental question related to the feasibility of deterministic symmetry breaking in the infinite Euclidean plane for two robots that have minimal or no knowledge of the respective capabilities and "measuring instruments" of themselves and each other. Assume that two anonymous mobile robots are placed at different locations at unknown distance d from each other on the infinite Euclidean plane. Each robot knows neither the location of itself nor of the other robot. The robots cannot communicate wirelessly, but have a certain nonzero visibility radius r (with range r unknown to the robots). By rendezvous we mean that they are brought at distance at most r of each other by executing symmetric (identical) mobility algorithms. The robots are moving with unknown and constant but not necessarily identical speeds, their clocks and pedometers may be asymmetric, and their chirality inconsistent.

We demonstrate that rendezvous for two robots is feasible under the studied model iff the robots have either: different speeds; or different clocks; or different orientations but equal chiralities. When the rendezvous is feasible, we provide a universal algorithm which always solves rendezvous despite the fact that the robots have no knowledge of which among their respective parameters may be different.

1 Introduction

Two anonymous robots (represented as points) are placed at unknown but different locations on the infinite Euclidean plane. They can move with constant speeds (which are not guaranteed to be different), their clocks may be asymmetric (they do not run on the same time unit), the distance units they may be able to measure may be different, their orientation and chirality possibly inconsistent, and their initial Euclidean distance unknown to them. To make matters worse, the robots not only cannot communicate wirelessly, but also all of the above parameters, namely moving-speed, clock-speed, distance-unit, orientation, and chirality, are unknown to them. The only ability the robots have is limited visibility so they can see each other only if they are within a given range, albeit this is also unknown to them and its magnitude may not be related in any way to their initial distance.

The question arising in this setting is: can the two robots rendezvous? I.e., can they meet at the same point in the plane? It is not at all obvious that a solution could exist in the general setting described above, where knowledge of their moving-speeds, and the consistency of their clock-speeds, distance-units, orientations, and chiralities cannot be assured. In fact, it might even seem counter-intuitive that the problem could be solvable. But if it were, one would want to know what strategies should the robots employ so as to minimize their rendezvous time. Note that the robots operate in the infinite Euclidean plane on which each robot may move with its constant speed. We are interested in symmetric rendezvous so that the robots must execute the same strategy as opposed to the corresponding asymmetric rendezvous problem which may have an optimal solution if one robot waits at its original location while the other is searching for it.

The fundamental problem in the question posed above is related to the feasibility of deterministic rendezvous. More broadly, one is interested to identify the parameters of the given model under which rendezvous is possible to achieve in finite time. Evidently, the overall concern of any strategy is how to break symmetry. However, unlike traditional ways of symmetry breaking where a robot can make use of *unconcealed* parameters, such as tokens, markers, white-boards, and labels, in our setting we are interested in the possibility of rendezvous under *unknown built-in attributes*, which include parameters important to the operation of the robots, such as moving-speed, clock-speed, distance-unit, orientation, and chirality, which may not have to be revealed by the robots during the execution of the rendezvous algorithm. Our overall objective is to design algorithms that prove the feasibility of rendezvous in such a constrained environment and whenever possible achieve good performance for the time spent by the robots to rendezvous.

In the present paper we show how two robots placed at unknown distance in the Euclidean plane can break symmetry when some of their unknown built-in attributes, including moving-speed, clock-speed, distance-unit, orientation, and chirality are different, and in some instances leading to a universal algorithm which guarantees rendezvous. Furthermore, and contrary to the case of rendezvous for robots with unconcealed parameters studied in the past, knowledge as to which of the parameters is different may be unnecessary and in any case may not even be given as input to the algorithm. Moreover our robots are completely unaware of the value(s) of their individual hidden parameters and do not make use of them in the computations needed to run the algorithm.

When no parameter of our scenario permits to break symmetry, the robots walk indefinitely and the rendezvous never takes place. However, as the robots have no knowledge of their parameters, it is obvious that they can never stop their walks and conclude that the rendezvous is indeed infeasible.

1.1 Model

We consider the symmetric rendezvous problem of two mobile robots $\mathcal{R}$ and $\mathcal{R}'$ modeled as points on the infinite Euclidean plane. The robots are initially located an unknown distance d from each other and each has a non-zero radius of visibility r. The rendezvous problem is solved the first time that the robots see each other, i.e. the first time they are a distance at most r from each other. The robots must employ the same algorithm in order to rendezvous. We assume that robots can store and compute real numbers with arbitrary precision.

We consider a model in which each robot has its own constant speed and in which each is equipped with a clock and compass allowing them to respectively measure their travel time and travel direction. We assume that robots can use their clocks to measure arbitrarily small time intervals. Each robot will consider itself as the origin of its own coordinate system and it will use its clock and compass to fix the distance unit (the product of its speed and local time unit) and orientation of its coordinate axes. We explicitly consider the possibility that the robots have different speeds, clocks, and/or compasses. We study algorithms which progress in a synchronous and continuous time model (i.e. robots are always active) and where robots can be instructed to move to any real position on the plane.

Without loss of generality, we will present our analysis from the viewpoint of the robot $\mathcal{R}$ and thus assume that this robot has maximum unit speed, and that its clock and compass are "correct" in the sense that they agree with some predefined global coordinate system. On the other hand, we set the speed of $\mathcal{R}'$ as $v > 0$; its time unit as $\tau > 0$; its orientation as $\phi \in [0, 2\pi)$; and its chirality as $\chi = \pm 1$. The overall effect of these differences in *reference frames* is that the robots will follow different trajectories despite them using the same algorithm. More specifically, $v \neq 1$ implies the robots have different speeds and will therefore travel different distances in the same unit of time; $\tau \neq 1$ implies that one time unit as measured by the clock of $\mathcal{R}'$ will in fact be τ time units as measured by that of $\mathcal{R}$; $\phi \in [0, 2\pi)$ implies that the coordinate axes of $\mathcal{R}'$ have been rotated (counter-clockwise) by an angle ϕ with respect to those of $\mathcal{R}$; and $\chi = -1$ implies that $\mathcal{R}'$ and $\mathcal{R}$ disagree on the $+y$ direction.

1.2 Related work

Rendezvous problems are well-known in the scientific literature. They are not only of theoretical interest due to the fundamental challenges one encounters to provide adequate mathematical solutions. They are also encountered in numerous applications which include the fields of operations research, search and rescue operations and planning in the mathematical sciences, as well as process synchronization, operating system design, and message sharing in computer science.

The rendezvous problem was first introduced informally in 1976 by Steve Alpern in [2] who later also formulated and formalized the continuous time version of the problem in [1]. A further impetus to the problem was given by the seminal book treatment [3] where rendezvous was viewed as a search-game between two players having the converging goal in that they are aiming to find one another as quickly as possible.

Numerous papers on rendezvous followed, covering several cases depending on various parameters of the model: type of environment (graph or geometric), robot's knowledge about the environment (partial or complete), anonymity of the robot (labeled or not), robot movement mode (synchronous, semi-synchronous or asynchronous), algorithm type (deterministic or randomized), reliability issues related to robot instruments, etc. A survey covering deterministic rendezvous algorithms is presented in [26] while the monograph [23] is dedicated to the ring and torus.

Many papers on rendezvous adopt the discrete model, e.g., where the robots may meet only at graph nodes (e.g., [27]). In the continuous model for graphs, it is possible to consider the agents' meeting in the interior of graph edges (e.g., [16]). However, when the continuous environment is a two-dimensional plane it is necessary to equip robots with devices permitting non-zero visibility, i.e. the rendezvous arises when the robots belong to each other's visibility range (e.g., see [4]).

Our formulation and analysis of rendezvous is based on the continuous time model in which robots are always active. Recent studies in this model include [22, 24] in which the authors focus on distributed local protocols concerning swarms that result in formations like "gathering at one point". In [12] gathering at a point in the plane is analyzed when some of the robots may be faulty. They design algorithms which achieve gathering of all reliable robots within the smallest possible time. In fact, they minimize the competitive ratio, defined as the ratio of the time required to achieve gathering of the reliable robots, to the time required for such gathering to occur under the assumption that the reliable robots were known in advance.

The rendezvous (and its more general version of gathering) problem has also been studied for robots with different speeds [6, 17], inconsistent compasses [8, 20] and chirality or sense of direction [5, 7]. In [19] the authors study the feasibility of gathering by mobile robots that have ϕ-absolute error dynamic compasses, which allows the angle difference between a local coordinate system and the global coordinate system to vary with time in the range of $[0, \phi]$. In [21] a gathering problem is discussed for robots equipped with inaccurate (incorrect) compasses which may point a different direction from other robots' compasses. However, in the studies previously mentioned, these differences were obstacles that needed to be circumvented by the suggested algorithms, rather than used for the benefit of the proposed approach, which is the case of the present paper. To the best of our knowledge, rendezvous in the plane for two robots with unknown attributes in continuous time has never been studied before.

The fundamental issue of the rendezvous problem is *symmetry breaking*, for example by exploiting some specific parameter(s) of the model that permit robots to act differently and not to be trapped in the same relative position to one another (cf. [26]). If there is no such parameter available it may be shown that the rendezvous is infeasible (e.g., [15]). For rendezvous in graph environments, the symmetry may be broken using asymmetry in the graph topology or the robots' positions within it (see [13]). In the case of rendezvous in the two-dimensional plane this is not possible and a symmetry-breaking procedure may exploit, for example, the difference of the robots labels [14, 16] or robot's knowledge of its own position in the Cartesian plane [9]. In all these cases the designer of the rendezvous algorithm needs to know what is this parameter of the studied scenario that permits the symmetry breaking. In the present paper it is shown that the knowledge, which of the parameters of the studied scenario makes the rendezvous possible, is not necessary.

Closely related to our current work is [11] which considers rendezvous of two anonymous robots on an infinite line when their walking speed and time units (which are unknown to the robots) may or may not be the same. The authors introduced the new concept of asymmetric clock and proposed a universal algorithm, so that the robots rendezvous in finite time, in any case when at least one of the parameters is not identical for the two robots.

1.3 Results and outline

The structure and results of the paper are as follows. In Section 2 we consider the problem of search and provide algorithms solving this problem nearly optimally. These algorithms will then form the basis for our rendezvous algorithms. In Section 3 we study the rendezvous problem under the assumption that the robots' time-units are equal. We demonstrate in this setting that rendezvous is always feasible if the robots differ in their speeds, and is only feasible when $v = 1$ if also $\chi = 1$ but $0 < \phi < 2\pi$. When the rendezvous is feasible we provide an algorithm which solves rendezvous in nearly optimal time.

In Section 4 we study the rendezvous problem when the robots' time units differ. In this case we demonstrate that rendezvous is always feasible when $\tau \neq 1$. Moreover, we provide an algorithm which universally solves rendezvous whenever the parameters of the robots are such that rendezvous is feasible. This algorithm does not require the robots to know which of their parameters differ. In Section 5 we conclude with a short discussion.

2 Search

The problem of search is as follows. We have a single robot $\mathcal{R}$ with radius of visibility $r > 0$ which needs to find a stationary target (exit, treasure, etc.) that is initially at an unknown distance d from the robot. This problem was solved in [25] where it was shown that the search time was in $\mathcal{O}\left(\log\left(\frac{d}{r}\right)\frac{d^2}{r}\right)$ and that this search time is optimal. We give a slightly different algorithm to the one in [25] which solves the problem in $\mathcal{O}\left(\log\left(\frac{d^2}{r}\right)\frac{d^2}{r}\right)$. We will use this search algorithm to build our rendezvous algorithms.

We construct our search algorithm using a number of smaller procedures/algorithms. The first two of these procedures are called $\texttt{SearchCircle}\,(\delta)$ and $\texttt{SearchAnnulus}\,(\delta_1, \delta_2, \rho)$. The procedure $\texttt{SearchCircle}\,(\delta)$ takes as input a positive real δ and instructs a robot to move from its initial position to a radius δ, move along the circle of radius δ, and return to its initial position. The procedure $\texttt{SearchAnnulus}\,(\delta_1, \delta_2, \rho)$ takes three positive real parameters δ_1, δ_2, and ρ. It repeatedly calls $\texttt{SearchCircle}\,(\cdot)$ with the end result that all points within the annulus of inner and outer radii δ_1 and δ_2 respectively have been within a distance at most ρ from the robot at some time during the algorithm. These procedures are formally described as Algorithms 1 and 2 respectively.

Algorithm 1 $\texttt{SearchCircle}\,(\delta)$

Input: $\delta > 0$ (real);
Begin:
1: Move along x-axis to radial position δ.
2: Traverse circle with radius δ.
3: Return to initial position.
:End

Algorithm 2 $\texttt{SearchAnnulus}\,(\delta_1, \delta_2, \rho)$

Input: $\delta_1 \geq 0$ (real); $\delta_2 > \delta_1$ (real); $\rho > 0$ (real);
Begin:
1: **for** $i = 0$ to $\left\lceil \frac{\delta_2 - \delta_1}{2\rho} \right\rceil$ **do** $\texttt{SearchCircle}\,(\delta_1 + 2i\rho)$.
:End

The next procedure $\texttt{Search}\,(k)$ takes a positive integer $k \geq 1$ as input and instructs a robot to search the set of $2k - 1$ annuli in such a way that: a) the j^{th}, $0 \leq j \leq 2k - 1$, annulus has inner and outer radii $\delta_{j,k} = 2^{-k+j}$ and $\delta_{j,k+1} = 2^{-k+j+1}$ respectively, and, b) all points of this annulus are approached within a distance of $\rho_{j,k} = 2^{-3k+2j-1}$. The idea is that the robot will search successive annuli of increasing inner and outer radii $\delta_{j,k}$ and $\delta_{j,k+1}$ under the assumption that its visibility radius is $\rho_{j,k}$. The specific values of $\delta_{j,k}$ and $\rho_{j,k}$ are chosen such that the ratio $\delta^2_{j,k}/\rho_{j,k} = 2^{k+1}$. At the end of the algorithm a robot is instructed to wait a rather specific amount of time only in order to simplify algebra later on. This procedure is formally described as Algorithm 3.

Algorithm 3 $\texttt{Search}\,(k)$

Input: $k > 0$ (integer);
Begin:
1: **for** $j = 0$ to $2k - 1$ **do**
2: $\quad\texttt{SearchAnnulus}\left(2^{-k+j}, 2^{-k+j+1}, 2^{-3k+2j-1}\right)$
3: Wait at initial position for a time $3(\pi + 1)(2^k + 2^{-k})$.
:End

We are now ready to introduce our main search algorithm. This algorithm is formally presented as Algorithm 4. The algorithm repeatedly runs $\texttt{Search}\,(k)$ until the target is discovered. The idea is that the robot will search regions of the plane in such a way that in each round k a robot spends time proportional to the ratio d^2/r under the assumption that the values of d and r were such that the target was discovered on the round k.

Algorithm 4

Begin:

```
1: k ← 1;
2: repeat
3:     Perform Search (k);
4:     k ← k + 1;
5: until Target found.
```

:End

We now claim the following:

Theorem 1. *Algorithm 4 solves the search problem in time*

$$T(d,r) < 6(\pi+1)\log\left(\frac{d^2}{r}\right)\frac{d^2}{r}.$$

We will prove this theorem using a series of lemmas. We begin by demonstrating that the algorithm is correct.

Lemma 1. *Algorithm 4 solves the search problem.*

Proof. Assume that the target is at distance d and the robot has visibility r. In the round k and sub-round j of Algorithm 3 a robot will search the entire disk of radius at least 2^{-k+j+1} with a granularity of at most $2^{-3k+2j-1}$. The robot must therefore find the target by the end of the round k of Algorithm 4 for which there exists an integer j such that $0 \le j \le 2k-1$, $2^{-k+j+1} \ge d$, and $2^{-3k+2j-1} \le r$. It is not hard to confirm, for example, that $k = \left\lfloor \log \frac{d^2}{r} \right\rfloor$ and $j = \lfloor \log(d) \rfloor + \left\lfloor \log \frac{d^2}{r} \right\rfloor$ satisfy these constraints. □

We now compute the running times of Algorithms 1 - 4.

Lemma 2. *It takes time:*

- $2(\pi+1)\delta$ *to complete* `SearchCircle` (δ).
- $2(\pi+1)\left(1+\left\lceil\frac{\delta_2-\delta_1}{2\rho}\right\rceil\right)\left(\delta_1+\rho\left\lceil\frac{\delta_2-\delta_1}{2\rho}\right\rceil\right)$ *to complete* `SearchAnnulus` (δ_1,δ_2,ρ).
- $3(\pi+1)(k+1)\cdot 2^{k+1}$ *to complete* `Search` (k).
- $3(\pi+1)k\cdot 2^{k+2}$ *to complete the first k rounds of Algorithm 4.*

Proof. `SearchCircle` (δ) clearly takes time $2(\pi+1)\delta$. Let $m = \left\lceil\frac{\delta_2-\delta_1}{2\rho}\right\rceil$. The time to complete `SearchAnnulus` (δ_1,δ_2,ρ) is then

$$\begin{aligned}\sum_{i=0}^{m} 2(\pi+1)(\delta_1+2i\rho) &= 2(\pi+1)\left[(m+1)\delta_1+\rho(m+1)m\right]\\ &= 2(\pi+1)\left(1+m\right)\left(\delta_1+\rho m\right).\end{aligned}$$

Now consider the algorithm `Search` (k). The time to complete the round j of this algorithm is just the time required to complete `SearchAnnulus` $(\delta_{j,k},\delta_{j,k+1},\rho_{j,k})$ where $\delta_{j,k} = 2^{-k+j}$ and $\rho_{j,k} = 2^{-3k+2j-1}$. Observe that $\frac{\delta_{j+1,k}-\delta_{j,k}}{2\rho_{j,k}} = \frac{2^{-k+j+1}-2^{-k+j}}{2^{-3k+2j}} = 2^{2k-j}$. The time to complete one round of the algorithm is then $2(\pi+1)\left(1+2^{2k-j}\right)\left(2^{-k+j}+2^{-3k+2j-1}\cdot 2^{2k-j}\right)$ which simplifies to $3(\pi+1)(2^{j-k}+2^k)$. Since

$$3(\pi+1)\sum_{j=0}^{2k-1}(2^{j-k}+2^k) = 3(\pi+1)\left[k2^{k+1}+2^{-k}(2^{2k}-1)\right]$$

we can conclude that the time required to complete `Search` (k) is

$$\begin{aligned}3(\pi+1)(2^k+2^{-k}) &+ 3(\pi+1)\left[k2^{k+1}+2^{-k}(2^{2k}-1)\right]\\ &= 3(\pi+1)(k+1)\cdot 2^{k+1}.\end{aligned}$$

Finally, the time to complete the first k rounds of Algorithm 4 is $3(\pi+1)\sum_{j=1}^{k}(j+1)2^{j+1} = 3(\pi+1)k\cdot 2^{k+2}$. $\square$

The next lemma gives a lower bound on the value of $\frac{d^2}{r}$ assuming that the target was found on round k of Algorithm 4:

Lemma 3. *If a robot finds the target on the round k of Algorithm 4 then $\frac{d^2}{r} \geq 2^{k+1}$.*

Proof. Assume that the target is found on round k and sub-round j, $0 \leq j \leq 2k-1$. In this case we know that $d \geq \delta_{j,k} = 2^{-k+j}$ and $r \leq \rho_{j,k} = 2^{-3k+2j-1}$. We therefore have that $\frac{d^2}{r} \geq \frac{\delta_{j,k}^2}{\rho_{j,k}} = 2^{k+1}$. $\square$

The proof of Theorem 1 is now simple:

Proof. (Theorem 1) By Lemma 1 there must exist a round during which the robot finds the target. Assume that the target is found on the round k. Then, by Lemma 3, we have $\frac{d^2}{r} \geq 2^{k+1}$ and the time to find the target is upper-bounded by the time to complete the first k rounds of Algorithm 4. Thus, if $T(d,r)$ is the rendezvous time, then by Lemma 2 we have that

$$T(d,r) \leq 3(\pi+1)k\cdot 2^{k+2} \leq 6(\pi+1)\left[\log\left(\frac{d^2}{r}\right)-1\right]\frac{d^2}{r}.$$

$\square$

3 Rendezvous with symmetric clocks

In this section we consider the rendezvous problem when the time units of the two robots are equal, i.e. $\tau = 1$. We will see that this problem is intimately related to search. Indeed, our goal is to demonstrate that the same Algorithm 4 used for search also solves the rendezvous problem (whenever a solution exists).

Theorem 2. *Algorithm 4 solves the rendezvous problem for two robots with equal time units in time*

$$T(d,r,v,\phi,\chi) < \begin{cases} 6(\pi+1)\log\left(\frac{d^2}{\mu r}\right)\frac{d^2}{\mu r}, & \chi = 1\\ 6(\pi+1)\log\left(\frac{d^2}{(1-v)r}\right)\frac{d^2}{(1-v)r}, & \chi = -1\end{cases}$$

where $\mu = \sqrt{v^2-2v\cos(\phi)+1}$.

To prove this theorem we will demonstrate that any algorithm which solves rendezvous for two robots with visibility r and initial distance d necessarily generates a corresponding algorithm which solves an instance of the search problem for a single robot with visibility r and initial distance d from an unknown target. The theorem will follow by analyzing the corresponding search algorithm when we use Algorithm 4 as our rendezvous algorithm.

We observe that we can interpret an algorithm for either the search or rendezvous problems as a single parametric trajectory $\vec{\mathcal{S}}(t)$ which specifies how the robots should move. In the case of rendezvous things are somewhat complicated by the fact that we have two robots which may not necessarily agree on their *reference frames*, i.e. they may have different speeds, orientations, and/or chiralities. Thus, although only a single trajectory is specified by an algorithm, the two robots will actually follow two different trajectories during the execution of that algorithm – $\mathcal{R}$ will follow the trajectory $\vec{\mathcal{S}}(t)$ and $\mathcal{R}'$ will follow a modified trajectory $\vec{\mathcal{S}}'(t)$ which depends on the specific values of v, ϕ, and χ. Nevertheless, the single trajectory $\vec{\mathcal{S}}(t)$ suffices to completely describe a rendezvous algorithm. In the sequel we will refer to rendezvous and/or search algorithms by the trajectory $\vec{\mathcal{S}}(t)$ they correspond to.

In the case of search, an algorithm $\vec{\mathcal{S}}(t)$ solves the problem if there exists a time $t \geq 0$ for which $|\vec{\mathcal{S}}(t) - \vec{d}| \leq r$ with $\vec{d}$, $|\vec{d}| = d$, a vector pointing from the initial position of the robot to the target. In the case of rendezvous, $\vec{\mathcal{S}}(t)$ solves the problem if there exists a time $t \geq 0$ for which $|\vec{\mathcal{S}}(t) - \vec{\mathcal{S}}'(t) - \vec{d}| \leq r$, where, in this case, $\vec{d}$, $|\vec{d}| = d$, represents a vector pointing from the initial position of $\mathcal{R}$ to that of $\mathcal{R}'$. If we reinterpret the trajectory $\vec{\mathcal{S}}_\circ(t) = \vec{\mathcal{S}}(t) - \vec{\mathcal{S}}'(t)$ as a search algorithm, then it is easy to see that $\vec{\mathcal{S}}(t)$ solves rendezvous for a given d and r if (and only if) $\vec{\mathcal{S}}_\circ(t)$ solves search for the same d and r. Moreover, if $\vec{\mathcal{S}}(t)$ solves rendezvous at the time T, then $\vec{\mathcal{S}}_\circ(t)$ solves search at the same time T. Thus, we say that the rendezvous algorithm $\vec{\mathcal{S}}(t)$ induces an equivalent search algorithm $\vec{\mathcal{S}}(t) - \vec{\mathcal{S}}'(t)$. This observation provides our strategy to prove Theorem 2 – we show that Algorithm 4, taken as a rendezvous algorithm, induces an equivalent search algorithm which still solves the search problem.

To begin we need to express the trajectory $\vec{\mathcal{S}}'(t)$ of the robot $\mathcal{R}'$ in terms of v, ϕ, χ, and $\vec{\mathcal{S}}(t)$.

Lemma 4. *Consider a rendezvous algorithm $\vec{\mathcal{S}}(t)$. Then the robot $\mathcal{R}$ will follow the trajectory $\vec{\mathcal{S}}(t)$ and the robot $\mathcal{R}'$ will follow the trajectory $\vec{\mathcal{S}}'(t) + \vec{d}$ where*

$$\vec{\mathcal{S}}'(t) = \begin{bmatrix} v\cos(\phi) & -v\chi\sin(\phi) \\ v\sin(\phi) & v\chi\cos(\phi) \end{bmatrix} \vec{\mathcal{S}}(t)$$

and $\vec{d}$ points from the initial position of $\mathcal{R}$ to the initial position of $\mathcal{R}'$.

Proof. If the robots have different chiralities ($\chi = -1$) then they will disagree on the $+y$ direction. This implies that $\mathcal{R}'$ will move along a trajectory that is a reflection about the x-axis of the trajectory followed by $\mathcal{R}$. If the orientation of $\mathcal{R}'$ is ϕ, then its trajectory will be rotated by an angle ϕ with respect to that of $\mathcal{R}$. If the speed of $\mathcal{R}'$ is $v < 1$ then it will travel a smaller distance in the same time interval as compared to v and its trajectory will thus be scaled by a factor of v. Combining all of these transformations together we find that $\vec{\mathcal{S}}'(t) = v \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \chi \end{bmatrix} \vec{\mathcal{S}}(t)$. $\square$

Now that we know the trajectory of $\mathcal{R}'$ we can compute the equivalent search trajectory $\vec{\mathcal{S}}_\circ(t) = \vec{\mathcal{S}}(t) - \vec{\mathcal{S}}'(t)$ as

$$\vec{\mathcal{S}}_\circ(t) = \vec{\mathcal{S}}(t) - \vec{\mathcal{S}}'(t) = \begin{bmatrix} 1 - v\cos(\phi) & v\chi\sin(\phi) \\ -v\sin(\phi) & 1 - v\chi\cos(\phi) \end{bmatrix} \vec{\mathcal{S}}(t).$$

Let $\mathbf{T}_\circ = \begin{bmatrix} 1 - v\cos(\phi) & v\chi\sin(\phi) \\ -v\sin(\phi) & 1 - v\chi\cos(\phi) \end{bmatrix}$. In the next lemma we rewrite $\mathbf{T}_\circ$ in a more convenient form.

Lemma 5. *The matrix $\mathbf{T}_\circ$ can be factored as $\mathbf{T}_\circ = \mathbf{\Phi}\mathbf{T}'_\circ$ where $\mathbf{\Phi}$ is a rotation matrix, $\mathbf{T}'_\circ = \begin{bmatrix} \mu & \frac{-(1-\chi)v\sin(\phi)}{\mu} \\ 0 & \frac{\chi v^2 - (1+\chi)v\cos(\phi) + 1}{\mu} \end{bmatrix}$ and $\mu = \sqrt{v^2 - 2v\cos(\phi) + 1}$.*

Proof. What we want amounts to a QR-factorization. This is a well known operation and so we just quote the result:

$$\mathbf{T}_\circ = \frac{1}{\mu} \begin{bmatrix} 1 - v\cos(\phi) & v\sin(\phi) \\ -v\sin(\phi) & 1 - v\cos(\phi) \end{bmatrix} \begin{bmatrix} \mu & \frac{-(1-\chi)v\sin(\phi)}{\mu} \\ 0 & \frac{1 + \chi v^2 - (1+\chi)v\cos(\phi)}{\mu} \end{bmatrix}$$

where we have set $\mu = \sqrt{v^2 - 2v\cos(\phi) + 1}$. It is easy to confirm that $\frac{1}{\mu} \begin{bmatrix} 1 - v\cos(\phi) & v\sin(\phi) \\ -v\sin(\phi) & 1 - v\cos(\phi) \end{bmatrix}$ is an orthogonal matrix with determinant 1. $\square$

Since rotations do not scale distances, the condition $|\mathbf{T}_\circ \vec{\mathcal{S}}(t) - \vec{d}| \leq r$ is equivalent to the condition that $|\mathbf{T}'_\circ \vec{\mathcal{S}}(t) - \vec{d}| \leq r$. It will be easier to analyze this rotated version of the problem and thus, in the sequel, we will use the following definition for an equivalent search trajectory:

Definition 1. *The equivalent search trajectory induced by the rendezvous trajectory $\vec{\mathcal{S}}(t)$ is the trajectory $\vec{\mathcal{S}}_\circ(t)$ where*

$$\vec{\mathcal{S}}_\circ(t) = \mathbf{T}_\circ \vec{\mathcal{S}}(t) = \begin{bmatrix} \mu & \frac{-(1-\chi)v\sin(\phi)}{\mu} \\ 0 & \frac{1+\chi v^2-(1+\chi)v\cos(\phi)}{\mu} \end{bmatrix} \vec{\mathcal{S}}(t)$$

and $\mu = \sqrt{v^2 - 2v\cos(\phi) + 1}$.

At this point it will be easier to consider the cases that $\chi = \pm 1$ separately. We begin with the case that $\chi = 1$:

Lemma 6. *Algorithm 4 solves rendezvous in time*

$$T(d, r, v, \phi) < 6(\pi + 1)\log\left(\frac{d^2}{\mu r}\right)\frac{d^2}{\mu r}$$

where $\mu = \sqrt{v^2 - 2v\cos(\phi) + 1}$.

Proof. We first observe that $\mathbf{T}_\circ$ takes on a particularly simple form when $\chi = 1$. We find that

$$\mathbf{T}_\circ = \begin{bmatrix} \mu & \frac{-(1-\chi)v\sin(\phi)}{\mu} \\ 0 & \frac{1+\chi v^2-(1+\chi)v\cos(\phi)}{\mu} \end{bmatrix} = \begin{bmatrix} \mu & 0 \\ 0 & \mu \end{bmatrix}$$

with the result that $\vec{\mathcal{S}}_\circ(t) = \mu\vec{\mathcal{S}}(t)$. Thus $\vec{\mathcal{S}}_\circ(t)$ is the trajectory of a robot with speed μ performing Algorithm 4. We need to demonstrate that there exists a time $t \geq 0$ for which $|\mu\vec{\mathcal{S}}(t) - \vec{d}| \leq r$. Multiplying this inequality by $\frac{1}{\mu}$ gives the equivalent condition $|\vec{\mathcal{S}}(t) - \frac{\vec{d}}{\mu}| \leq \frac{r}{\mu}$ and we already know that there exists a time for which this is satisfied when $\vec{\mathcal{S}}(t)$ is given by Algorithm 4. We can therefore conclude that Algorithm 4 solves rendezvous. To bound the rendezvous time we can directly use the results of Theorem 1 with d and r replaced with $\frac{d}{\mu}$ and $\frac{r}{\mu}$ respectively. □

Now we consider the case that the robots have opposite chiralities. We begin by revisiting what it means for an algorithm to solve the search problem. To this end, assume that we have an algorithm $\vec{\mathcal{S}}(t)$ that solves search in time at most T when the robot has visibility r and the target is at a distance d. What this tells us is that all possible positions of the target at distance d from the initial position of the robot can be reached in time at most T. In particular, if the target is located at $\vec{d}$, then there exists a time $0 \leq t \leq T$ for which $|\vec{\mathcal{S}}(t) - \vec{d}| \leq r$. Since the distance of closest approach of the robot and the target occurs when the robot is located on the line defined by the unit vector $\hat{d} = \vec{d}/d$, we can conclude that there exists a time $0 \leq t \leq T$ for which the length of the projection of $\vec{\mathcal{S}}(t) - \vec{d}$ onto the line defined by $\hat{d} = \vec{d}/d$ is at most r. Thus, if $\vec{\mathcal{S}}(t)$ solves search, then there exists a time $0 \leq t \leq T$ for which $(\vec{\mathcal{S}}(t) - \vec{d}) \cdot \hat{d} \leq r$. With this observation in hand, we can proceed to prove the following:

Lemma 7. *When $\chi = -1$ Algorithm 4 solves rendezvous in time*

$$T(d, r, v) < 6(\pi + 1)\log\left(\frac{d^2}{(1-v)r}\right)\frac{d^2}{(1-v)r}.$$

Proof. When $\chi = -1$ the matrix $\mathbf{T}_\circ$ simplifies to

$$\mathbf{T}_\circ = \begin{bmatrix} \mu & \frac{-2v\sin(\phi)}{\mu} \\ 0 & \frac{1-v^2}{\mu} \end{bmatrix}.$$

We want to show that there exists a time $t \geq 0$ for which $(\vec{\mathcal{S}}_\circ(t) - \vec{d}) \cdot \hat{d} \leq r$ is satisfied. Observe that

$$
\begin{aligned}
(\vec{\mathcal{S}}_\circ(t) - \vec{d}) \cdot \hat{d} &= (\mathbf{T}_\circ \vec{\mathcal{S}}(t)) \cdot \hat{d} - d = \vec{\mathcal{S}}^T(t)\mathbf{T}_\circ^T \hat{d} - d \\
&= \vec{\mathcal{S}}(t) \cdot (\mathbf{T}_\circ^T \hat{d}) - d = |\mathbf{T}_\circ^T \hat{d}| \left[\vec{\mathcal{S}}(t) \cdot \frac{\mathbf{T}_\circ^T \hat{d}}{|\mathbf{T}_\circ^T \hat{d}|} - \frac{d}{|\mathbf{T}_\circ \hat{d}|} \right] \\
&= |\mathbf{T}_\circ^T \hat{d}| \left[(\vec{\mathcal{S}}(t) - \vec{d'}) \cdot \hat{d}' \right]
\end{aligned}
$$

where $\hat{d}' = \frac{\mathbf{T}_\circ^T \hat{d}}{|\mathbf{T}_\circ^T \hat{d}|}$ and $\vec{d'} = \frac{d}{|\mathbf{T}_\circ^T \hat{d}|}\hat{d}'$. Thus, the condition that $(\vec{\mathcal{S}}_\circ(t) - \vec{d}) \cdot \hat{d} \leq r$ is satisfied for some $t \geq 0$ is equivalent to the condition that $(\vec{\mathcal{S}}(t) - \vec{d'}) \cdot \hat{d}' \leq r'$ with $r' = \frac{r}{|\mathbf{T}_\circ^T \hat{d}|}$. As we know that Algorithm 4 indeed does satisfy this equality for some $t \geq 0$ we can conclude that Algorithm 4 solves rendezvous. To bound the time of rendezvous we need to replace d and r with $\frac{d}{|\mathbf{T}_\circ^T \hat{d}|}$ and $\frac{r}{|\mathbf{T}_\circ^T \hat{d}|}$ in the bound of Theorem 1. To complete the proof we first need to compute the value of $|\mathbf{T}_\circ^T \hat{d}|$ and then maximize the resulting time bound over all possible positions of the target. As we only need to compute the worst case position of $\hat{d}$ in order to bound the rendezvous time, this justifies that we set $\vec{d} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and instead maximize the result over all possible values of ϕ. In this case it is easy to confirm that $|\mathbf{T}_\circ^T \hat{d}| = \frac{1-v^2}{\mu}$. Since the search time scales with $\frac{d^2}{r}$ we want to maximize the quantity $\frac{\mu}{1-v^2}$ with respect to ϕ. This is equivalent to maximizing $\mu = \sqrt{v^2 - 2v\cos(\phi) + 1}$. Clearly, the maximum value of μ is $\sqrt{v^2 + 2v + 1} = 1 + v$. The bound in the lemma then follows from Theorem 1 and the fact that $\frac{1+v}{1-v^2} = \frac{1}{1-v}$. $\square$

4 Rendezvous with asymmetric clocks

We now consider rendezvous for the case that the robots have different time units ($\tau \neq 1$). Without loss of generality we will take $\tau < 1$. We will begin with the assumption that the robots have the same speeds ($v = 1$) and then extend our results to cover the case that the speeds of the robots may also be different. Our goal is to demonstrate that rendezvous is always feasible. The main result of this section follows:

Theorem 3. *For any fixed $\tau < 1$ there exists an algorithm which solves rendezvous in finite time.*

We will prove this theorem constructively and give an algorithm which solves the rendezvous problem in finite time provided that $\tau < 1$. This algorithm will be constructed using the algorithms of Section 2. In the sequel we describe this algorithm and then analyze its running time.

An outline and the idea of our rendezvous algorithm is as follows. The algorithm proceeds in a series of rounds with each round composed of equal length inactive and active phases. In an inactive phase a robot will remain stationary at its initial position. In the active phase of round n a robot will (essentially) perform the first n rounds of Algorithm 4. We observe that this implies that there will be a round n_* of the algorithm for which, say, the robot $\mathcal{R}$ would find $\mathcal{R}'$ if $\mathcal{R}'$ were stationary at its initial position. Moreover, this will be true for every round $n \geq n_*$. We will show that, due to the differing time units of the robots, it will happen that the active and inactive phases of the robots overlap and that the length of the overlap interval grows without bound. Thus, there will exist a round $n \geq n_*$ during which the active and inactive phases of the robots overlap long enough that $\mathcal{R}$ is able to find $\mathcal{R}'$ waiting at its initial location.

Our rendezvous algorithm is formally described as Algorithm 7. In each active phase a robot will run the procedures `SearchAll` (n) and `SearchAllRev` (n) which are formally described as Algorithms 5 and 6 respectively. The procedure `SearchAll` (n) is identical to Algorithm 4 except that it always terminates after n rounds. The procedure `SearchAllRev` (n) is identical to `SearchAll` (n) except that it is run in reverse, i.e. it begins on the n^{th} round and ends on the round 1. Figures 1 and 2 illustrate the structure of each round of the algorithm.

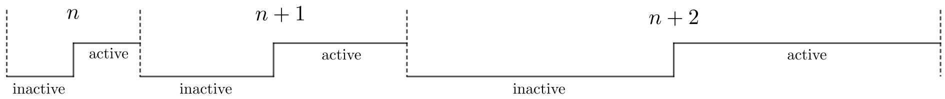

Figure 1: Three rounds of the algorithm.

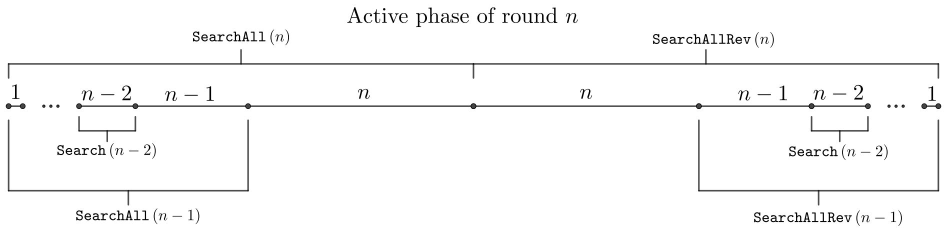

Figure 2: Structure of the active phase of round n.

Algorithm 5 `SearchAll` (n)

Input: $n > 0$ (integer);
Begin:
1: **for** $k = 1$ to n **do** Perform `Search` (k).
:End

Algorithm 6 `SearchAllRev` (n)

Input: $n > 0$ (integer);
Begin:
1: **for** $k = n$ to 1 **do** Perform `Search` (k).
:End

Algorithm 7

Begin:
1: $n \leftarrow 1$
2: **repeat**
3: Wait at initial position for a time $2S(n)$.
4: Perform `SearchAll` (n) then `SearchAllRev` (n).
5: $n = n + 1$.
6: **until** Rendezvous occurs
:End

We begin our proof of Theorem 3 with the following lemma which gives the time at which the inactive and active phases of Algorithm 7 begin.

Lemma 8. *The n^{th} inactive phase of Algorithm 7 begins at the time $I(n) = 24(\pi + 1)[(2n - 4) \cdot 2^n + 4]$ and the n^{th} active phase begins at the time $A(n) = 24(\pi + 1)[(3n - 4) \cdot 2^n + 4]$.*

Proof. It is easy to see that each round of Algorithm 7 lasts time $4S(n)$ where $S(n)$ is the time it takes to complete `SearchAll` (n). Using the results of Lemma 2 we have

$$S(n) = 12(\pi + 1)n \cdot 2^n. \tag{1}$$

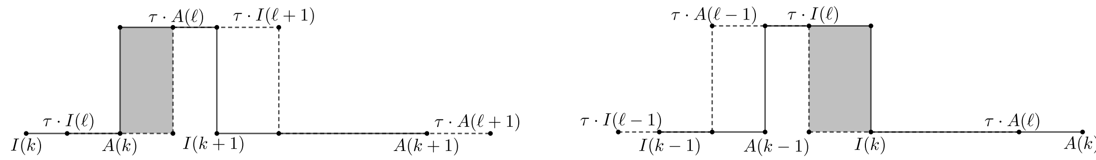

Figure 3: Illustration of how the active phase of $\mathcal{R}$ can overlap with the inactive phase of $\mathcal{R}'$. The intervals of overlap are shaded, the active/inactive phases of $\mathcal{R}$ are indicated by solid lines, and those of $\mathcal{R}'$ are indicated by dashed lines.

The beginning of the n^{th} inactive phase is just the total time required to complete the first $n-1$ rounds of the algorithm. Thus

$$I(n) = 4\sum_{k=1}^{n-1} S(k) = 48(\pi+1)\sum_{k=1}^{n-1} k2^k = 24(\pi+1)[(2n-4)2^n+4].$$

The n^{th} active phase begins when the n^{th} inactive phase ends. We therefore have $A(n) = I(n) + 2S(n) = 24(\pi + 1)[(3n-4)\cdot 2^n + 4]$. $\square$

To reiterate, we need to show that the overlap between the active and inactive phases of the robots grows without bound and there will therefore eventually be a round n_* of the algorithm for which $\mathcal{R}$ is able to finish the first (resp. last) k rounds of `SearchAll` (n_*) (resp. `SearchAllRev` (n_*)) during an inactive phase of $\mathcal{R}'$ where k is the first round of `SearchAll` $(\cdot)$ during which $\mathcal{R}$ would find $\mathcal{R}'$ if $\mathcal{R}'$ were stationary at its initial location. Since the robots run both of the procedures `SearchAll` (n_*) and `SearchAllRev` (n_*) the inactive and active phases of the robots can overlap in the two ways illustrated in Figure 3. In the next two lemmas we show under what circumstances the robots will find themselves in either of the two situations depicted in Figure 3.

Lemma 9. *If* $\frac{k}{(k+1+a)2^{a+1}} \le \tau \le \frac{3}{2}\cdot\frac{k}{(k+1+a)2^{a+1}}$ *and* $k \ge 2(a+1)$ *for some integer* $a \ge 0$ *then the* k^{th} *active phase of* $\mathcal{R}$ *overlaps with the* $(k+1+a)^{th}$ *inactive phase of* $\mathcal{R}'$ *by an amount* $\tau\cdot A(k+1+a) - A(k)$.

Proof. Referring to Figure 3a), one can see that in order for the k^{th} active phase of $\mathcal{R}$ to overlap with the $(k+a)^{th}$ inactive phase of $\mathcal{R}'$ by an amount $\tau\cdot A(k+1+a) - A(k)$ we need to satisfy $\tau\cdot I(k+a) \le A(k) \le \tau\cdot A(k+a)$. Rearranging these inequalities leads to the equivalent condition $\frac{A(k)}{A(k+1+a)} \le \tau \le \frac{A(k)}{I(k+1+a)}$. Thus, in order to prove the lemma we need to demonstrate that $\frac{A(k)}{A(k+1+a)} \le \frac{k}{(k+1+a)2^{a+1}}$ and $\frac{A(k)}{I(k+1+a)} \ge \frac{3}{2}\cdot\frac{k}{(k+1+a)2^{a+1}}$. Consider the latter inequality first. A simple rearrangement of this inequality yields $(k+1+a)2^{a+1}A(k) \le kA(k+1+a)$. Substitution of the expression derived in Lemma 8 for $A(k)$ gives

$$(k+1+a)2^{a+1}[(3k-4)2^k+4] \le k\{[3(k+1+a)-4]2^{k+1+a}+4\}.$$

After some manipulation we arrive to the equivalent condition $\frac{a+1}{k} > \frac{1-2^{-a-1}}{2^k-1}$. It is easy to confirm that this inequality is satisfied for all $a \ge 0$ and $k \ge 2(a+1)$.

Now consider the inequality $\frac{A(k)}{I(k+1+a)} \ge \frac{3}{2}\cdot\frac{k}{(k+1+a)2^{a+1}}$. Rearranging this inequality and substitution of the expressions for $A(k)$ and $I(k)$ from Lemma 8 yields

$$2(k+1+a)2^{a+1}[(3k-4)2^k+4] \le 3k\{[2(k+1+a)-4]2^{k+1+a}+4\}.$$

After some manipulation this reduces to $\frac{k}{a+1} \ge \frac{2}{1+3\frac{1-2^{-a-1}}{2^k-1}}$. It is easy to see that this inequality is also satisfied whenever $a \ge 0$ and $k \ge 2(a+1)$. $\square$

Lemma 10. *If* $\frac{2}{3}\cdot\frac{k}{(k+a)2^a} \le \tau \le \frac{k}{(k+1+a)2^a}$ *and* $k \ge 2(a+1)$ *for some integer* $a \ge 0$ *then the* $(k-1)^{st}$ *active phase of* $\mathcal{R}$ *overlaps with the* $(k+a)^{th}$ *inactive phase of* $\mathcal{R}'$ *by an amount* $I(k) - \tau\cdot I(k+a)$.

Proof. Referring to Figure 3b), in order for the $(k-1)^{st}$ active phase of $\mathcal{R}$ to overlap with the $(k+a)^{th}$ inactive phase of $\mathcal{R}'$ by an amount $I(k) - \tau \cdot I(k+a)$ we need to satisfy $\tau \cdot I(k+a) \leq I(k) \leq \tau \cdot A(k+a)$. Rearranging these inequalities leads to the equivalent condition $\frac{I(k)}{A(k+a)} \leq \tau \leq \frac{I(k)}{I(k+a)}$. Thus, in order to prove the lemma we need to demonstrate that $\frac{I(k)}{A(k+a)} \leq \frac{2}{3}\frac{k}{(k+a)2^a}$ and $\frac{I(k)}{I(k+a)} \geq \frac{k}{(k+1+a)2^a}$. Consider the latter inequality first. Rearrangment and substitution of the expressions for $A(k)$ and $I(k)$ gives

$$3(k+a)2^a[(2k-4)2^k+4] \leq 2k\{[3(k+a)-4]2^{k+a}+4\}.$$

After some manipulation we get $3a(2^k-1)+k(2^k-3+2^{1-a}) \geq 0$. It is easy to confirm that this inequality is satisfied for all $a \geq 0$ and $k \geq 2(a+1)$.

Now consider the inequality $\frac{I(k)}{I(k+a)} \geq \frac{k}{(k+1+a)2^a}$. Rearrangement and substitution of the expression for $I(k)$ yields

$$(k+1+a)2^a[(2k-4)2^k+4] \leq k\{[2(k+a)-4]2^{k+a}+4\}.$$

After some manipulation this reduces to $\frac{k}{a+1} \geq \frac{2}{1+\frac{3-2^{-a}}{2^k-1}}$. It is easy to see that this inequality is also satisfied whenever $a \geq 0$ and $k \geq 2(a+1)$. □

In the next two lemmas we determine in which round the robots will rendezvous under the assumption that the conditions of the previous two lemmas are met.

Lemma 11. *Assume that $\mathcal{R}$ would find a stationary target located at the initial position of $\mathcal{R}'$ on round n of Algorithm 7. If there exists two integers $a \geq 0$ and $k_0 \geq 2(a+1)$ such that τ satisfies $\frac{k}{(k+1+a)2^{a+1}} \leq \tau \leq \frac{3}{2} \cdot \frac{k}{(k+1+a)2^{a+1}}$ for all rounds $k \geq k_0$ then the robots will rendezvous by the end of the round $k_* = n + \left\lceil \log\left(\frac{n}{a+1}\right)\right\rceil$ of Algorithm 7.*

Proof. Assume that integers $a \geq 0$ and $k_0 \geq 2(a+1)$ exist such that τ satisfies the condition of the lemma. This implies that τ must satisfy

$$\frac{1}{2} \cdot 2^{-a} \leq \tau \leq \frac{3}{4} \cdot \frac{k_0}{k_0+1+a} \cdot 2^{-a} \tag{2}$$

since $\frac{k}{k+1+a}$ is increasing with k and $\frac{k}{k+1+a} < 1$.

By Lemma 9 the k^{th} active phase of $\mathcal{R}$ overlaps with the $(k+1+a)^{th}$ inactive phase of $\mathcal{R}'$ by an amount $\tau \cdot A(k+1+a) - A(k)$ for all $k \geq k_0$. The robots are guaranteed to rendezvous on or before the first round $k_* \geq k_0$ for which this overlap is larger than $S(n)$. We observe that

$$\begin{aligned}
&\tau A(k+1+a) - A(k)\\
&= 24(\pi+1)\left[\tau[(3(k+1+a)-4)2^{k+1+a}+4] - (3k-4)2^k - 4\right]\\
&= 24(\pi+1)\left[(3k-4)(\tau 2^{1+a}-1)2^k + 3\tau(a+1)2^{k+1+a} - 4(1-\tau)\right].
\end{aligned}$$

For τ satisfying (2) we can write

$$\begin{aligned}
\tau \cdot A(k+1+a) - A(k) &\geq 24(\pi+1)\left[3(a+1)2^k - 4\left(1-\frac{1}{2}2^{-a}\right)\right]\\
&> 24(\pi+1)\left[3(a+1)2^k - 4\right].
\end{aligned}$$

Thus we will have $\tau \cdot A(k+1+a) - A(k) \geq S(n)$ when $24(\pi+1)\left[3(a+1)\cdot 2^k - 4\right] \geq S(n)$. Using the expression (1) for $S(n)$ we can rewrite the above inequality as $3(a+1)\cdot 2^k - 4 \geq \frac{n}{2}\cdot 2^n$ which simplifies to $k \geq \log\left[\frac{1}{3(a+1)}\left(\frac{n}{2}\cdot 2^n + 4\right)\right]$. The robots will therefore rendezvous by the end of the round

$$k_* = \left\lceil \log\left[\frac{1}{3(a+1)}\left(\frac{n}{2}\cdot 2^n+4\right)\right]\right\rceil < n + \left\lceil \log\left(\frac{n}{a+1}\right)\right\rceil.$$

□

Lemma 12. *Assume that $\mathcal{R}$ would find a stationary target located at the initial position of $\mathcal{R}'$ on round n of Algorithm 7. If there exists two integers $a \geq 0$ and $k_0 \geq 2(a+1)$ such that τ satisfies $\frac{2}{3} \cdot \frac{k}{(k+a)2^a} \leq \tau \leq \frac{k}{(k+1+a)2^a}$ for all $k \geq k_0$ then the robots will rendezvous by the end of the round $k_* = n + \left\lceil \log(n) + \log\left(1 + \frac{k_0}{a+1}\right)\right\rceil$ of Algorithm 7.*

Proof. Assume that integers $a \geq 0$ and $k_0 \geq 2(a+1)$ exist such that τ satisfies the condition of the lemma. This implies that τ must satisfy

$$\frac{2}{3} \cdot 2^{-a} \leq \tau \leq \frac{k_0}{k_0+1+a} \cdot 2^{-a} \tag{3}$$

since $\frac{k}{k+1+a} < 1$ and is increasing with k.

Assume d and r are chosen such that $\mathcal{R}$ would find a stationary target at the initial position of $\mathcal{R}'$ on round n of Algorithm 7. By Lemma 10 the $(k-1)^{st}$ active phase of $\mathcal{R}$ overlaps with the $(k+a)^{th}$ inactive phase of $\mathcal{R}'$ by an amount $I(k) - \tau \cdot I(k+a)$ for all $k \geq k_0$. The robots are guaranteed to rendezvous on or before the first round $k_* \geq k_0$ for which this overlap is larger than $S(n)$. We observe that

$$\begin{aligned} &I(k) - \tau \cdot I(k+a) \\ &\quad = 24(\pi+1)\left[(2k-4)2^k + 4 - \tau[(2(k+a)-4)2^{k+a} + 4]\right] \\ &\quad = 24(\pi+1)\left[(2k-4)2^k[1-\tau 2^a] + 4(1-\tau) - 2a\tau 2^{k+a}\right]. \end{aligned}$$

Let $\gamma = \frac{k_0}{k_0+1+a}$. Then, for τ satisfying (3), we can write

$$\begin{aligned} &I(k) - \tau \cdot I(k+a) \\ &\quad \geq 24(\pi+1)\left[(2k-4)2^k(1-\gamma) + 4(1-\gamma 2^{-a}) - 2a\gamma 2^k\right] \\ &\quad > 24(\pi+1)\left[(2k-4)2^k(1-\gamma) - 2a\gamma 2^k\right] \\ &\quad = 48(\pi+1)[(k-2)(1-\gamma) - a\gamma]2^k \end{aligned}$$

Thus we will have $I(k) - \tau \cdot I(k+a) \geq S(n)$ when $[(k-2)(1-\gamma) - a\gamma]2^k \geq \frac{n}{4} \cdot 2^n$. Let $x = (k-2)(1-\gamma) - a\gamma$ such that $k = \frac{x+a\gamma}{1-\gamma} + 2 = \frac{x+(a-2)\gamma+2}{1-\gamma}$. We may write

$$\begin{aligned} [(k-2)(1-\gamma) - a\gamma]2^k &= x \cdot \left(2^{\frac{1}{1-\gamma}}\right)^x \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{(a-2)\gamma+2} \\ &= x \cdot e^{\frac{\ln(2)x}{1-\gamma}} \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{(a-2)\gamma+2}. \end{aligned}$$

Then, in terms of x, we need to satisfy,

$$x \cdot e^{\frac{\ln(2)x}{1-\gamma}} \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{(a-2)\gamma+2} \geq \frac{n}{4} \cdot 2^n$$

or

$$\frac{\ln(2)x}{1-\gamma} \cdot e^{\frac{\ln(2)x}{1-\gamma}} \geq \frac{\ln(2)n}{4(1-\gamma)} \cdot 2^n \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{-(a-2)\gamma-2}.$$

In the case of equality, the above has the form $ze^z = y$ and this has the solution $z = W(y)$ where $W(\cdot)$ is the W-Lambert function [10]. We therefore have that

$$\frac{x\ln(2)}{1-\gamma} = W\left[\frac{\ln(2)n}{4(1-\gamma)} \cdot 2^n \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{-(a-2)\gamma-2}\right].$$

Expressing this again in terms of k we find that k must satisfy

$$k \geq 2 + \frac{a\gamma}{1-\gamma} + \frac{1}{\ln(2)}W\left[\frac{\ln(2)n}{4(1-\gamma)} \cdot 2^n \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{-(a-2)\gamma-2}\right]$$

which implies that

$$k_* = 2 + \left\lceil \frac{a\gamma}{1-\gamma} + \frac{1}{\ln(2)} W\left[\frac{\ln(2)n}{4(1-\gamma)} \cdot 2^n \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{-(a-2)\gamma-2}\right]\right\rceil.$$

Now, $W(x)$ behaves asymptotically as $\ln(x) - \ln(\ln(x))$ [18] and we may therefore simplify this expression to

$$\begin{aligned} k_* &\le 2 + \left\lceil \frac{a\gamma}{1-\gamma} + \frac{1}{\ln(2)} \ln\left[\frac{\ln(2)n}{4(1-\gamma)} \cdot 2^n \cdot \left(2^{\frac{1}{1-\gamma}}\right)^{-(a-2)\gamma-2}\right]\right\rceil \\ &< n + \left\lceil \frac{1}{\ln 2} \ln\left(\frac{n}{1-\gamma}\right)\right\rceil = n + \left\lceil \log\left(\frac{n}{1-\gamma}\right)\right\rceil. \end{aligned}$$

Finally, rewriting the above in terms of k_0 leads us to the desired result. ☐

We now determine explicitly an upper bound on the round in which the robots rendezvous.

Lemma 13. *Parameterize τ as $\tau = t \cdot 2^{-a}$ for an integer $a \ge 0$ and a real $t \in [\frac{1}{2}, 1)$. Then, if $\frac{1}{2} \le t \le \frac{2}{3}$ the robots will rendezvous before the end of the round $k_* = \max\left\{8(a+1),\ n + \left\lceil \log\left(\frac{n}{a+1}\right)\right\rceil\right\}$ and otherwise, for $\frac{2}{3} < t < 1$, the robots will rendezvous before the end of the round $k_* = \max\left\{(a+1)\frac{t}{1-t},\ n + \left\lceil \log\left(\frac{n}{1-t}\right)\right\rceil\right\}$.*

Proof. First note that we may always write τ uniquely as $t \cdot 2^{-a}$ by taking $a = \lfloor -\log(\tau)\rfloor - 1$ and $t = \frac{1}{2}$ if τ is a power of two, and otherwise taking $a = \lfloor -\log(\tau)\rfloor$ and $t = \tau \cdot 2^a$.

First assume that $\frac{1}{2} \le t \le \frac{2}{3}$. Then the first part of the lemma will follow from Lemma 11 if we can find a k_0 such that $\tau = t \cdot 2^{-a} \le \frac{3}{4}\frac{k_0}{k_0+1+a} \cdot 2^{-a}$. Solving this inequality we find that k_0 is given by $k_0 \ge 4(a+1)\frac{t}{3-4t}$. For $t \in [\frac{1}{2}, \frac{2}{3}]$ one can easily confirm that k_0 must be at least $8(a+1)$ in order to guarantee that $k_0 \ge 4(a+1)\frac{t}{3-4t}$. Thus, if $n + \left\lceil \log\left(\frac{n}{a+1}\right)\right\rceil \ge 8(a+1)$ the robots will rendezvous before the end of the round $n + \left\lceil \log\left(\frac{n}{a+1}\right)\right\rceil$ otherwise the robots will rendezvous at the end of the round $8(a+1)$.

Now assume that $\frac{2}{3} < \tau < 1$. Then the second part of the lemma will follow from Lemma 12 if we can determine a k_0 such that $\tau \le \frac{k_0}{k_0+1+a} \cdot 2^{-a}$, or, equivalently, $t \le \frac{k_0}{k_0+1+a}$. Solving this inequality we find that $k_0 = \frac{(a+1)t}{1-t}$. Thus, if $n + \left\lceil \log\left(1 + \frac{k_0}{a+1}\right)\right\rceil \ge \frac{t}{1-t}(a+1)$ the robots will rendezvous before the end of the round $n + \left\lceil \log\left(1 + \frac{k_0}{a+1}\right)\right\rceil = n + \left\lceil \log\left(\frac{1}{1-t}\right)\right\rceil$, and otherwise the robots will rendezvous by the end of the round $(a+1)\frac{t}{1-t}$. ☐

At this point we can almost prove Theorem 3. For now we will prove the following weaker statement:

Lemma 14. *For any fixed $\tau < 1$ there exists an algorithm which solves rendezvous in finite time if $v = 1$.*

Proof. Write $\tau = t \cdot 2^{-a}$ where t and a are defined as in the previous lemma. Assume that $\mathcal{R}$ would find a stationary target at the initial position of on round n of Algorithm 7. Then $\frac{d^2}{r} \ge 2^{n+1}$ and also $n \le \log\left(\frac{d^2}{r}\right) - 1$ (see Lemma 3). The robots will rendezvous by the round k_* of Algorithm 7 where k_* is given in Lemma 13. The total time to complete k_* rounds is $I(k_*) = 24(\pi+1)[(2k_* - 4)2^{k_*} + 4]$. Thus, the rendezvous time of the algorithm is $T(d, r, \tau) < 24(\pi+1)[(2k_* - 4)2^{k_*} + 4]$. We claim that, for any fixed value of $\tau < 1$, this bound is finite. Indeed, observe that k_* is only infinite if $t = 1$. However, t is strictly smaller than one if $\tau < 1$. We have thus demonstrated that Algorithm 7 solves rendezvous in finite time if $\tau < 1$ and $v = 1$. ☐

We now extend this result to cover the case that the robots have different speeds. This will conclude the proof of Theorem 3.

Proof. (Theorem 3) Observe that the speed of a robot does not affect the times at which its active and inactive phases begin and/or end. This implies that everything up to and including Lemma 13 applies directly to the case that $v \neq 1$. This already implies that there will be a finite number of rounds until the robots rendezvous. A time bound can be derived in a similar manner to the derivation of the previous lemma. ☐

We note that we have not needed to consider the chirality and/or orientations of the robots in our analysis due to the fact that our proof relied on one robot finding the other while the other was stationary. Thus Theorem 3 applies regardless of the robots' relative orientations and/or chiralities. Moreover, one can use the same techniques of Section 3 to show that Algorithm 7 will also solve the rendezvous problem if the robots have different speeds but equal time units. We can therefore conclude this section with the following theorem:

Theorem 4. *Algorithm 7 solves the rendezvous problem in finite time if* $\tau \neq 1$ *or* $v \neq 1$, *or* $\chi = 1$ *and* $0 < \phi < 2\pi$.

5 Conclusions

In this paper we addressed the fundamental problem of feasibility of deterministic rendezvous in the infinite Euclidean plane for two robots that have minimal or no knowledge of the respective capabilities and "measuring instruments" of themselves and each other. We examined the impact on feasibility of rendezvous that the four parameters of speed, clock, distance, chirality have and presented and analyzed specific algorithms with good performance guarantees on the rendezvous time.

Our approach not only provides a surprising twist to the well-known rendezvous problem on the infinite Euclidean plane, but possibly it also creates interesting avenues for future research. In addition to tightening our bounds, there are several interesting questions that could be considered concerning the rendezvous problem. These include rendezvous for robots that may have alternative capabilities (e.g., variable speed), more general terrains with and without obstacles, and rendezvous in higher dimensional space. In addition, it would be challenging to solve deterministic gathering for multiple robots in this setting of "minimal knowledge".

References

[1] S. Alpern. The rendezvous search problem. *SIAM Journal on Control and Optimization*, 33(3):673–683, 1995.

[2] S. Alpern. Hide and seek games, Seminar, Institut für Höhere Studien, Wien, 26 July, 1976.

[3] S. Alpern and S. Gal. *The theory of search games and rendezvous*, volume 55. Kluwer Academic Publishers, 2002.

[4] E. J. Anderson and S. P. Fekete. Two dimensional rendezvous search. *Operations Research*, 49(1):107–118, 2001.

[5] L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Rendezvous and election of mobile agents: Impact of sense of direction. *Theory Comput. Syst.*, 40(2):143–162, 2007.

[6] S. Bouchard, Y. Dieudonné, A. Pelc, and F. Petit. On deterministic rendezvous at a node of agents with arbitrary velocities. *Inf. Process. Lett.*, 133:39–43, 2018.

[7] Q. Bramas and S. Tixeuil. Wait-free gathering without chirality. In *Structural Information and Communication Complexity - 22nd International Colloquium, SIROCCO 2015, Montserrat, Spain, July 14-16, 2015, Post-Proceedings*, pages 313–327, 2015.

[8] R. Cohen and D. Peleg. Convergence of autonomous mobile robots with inaccurate sensors and movements. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 549–560. Springer, 2006.

[9] A. Collins, J. Czyzowicz, L. Gasieniec, and A. Labourel. Tell me where I am so I can meet you sooner. In *ICALP*, pages 502–514, 2010.

[10] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5(1):329–359, Dec 1996.

[11] J. Czyzowicz, R. Killick, and E. Kranakis. Linear rendezvous with asymmetric clocks. In *OPODIS Conference on Principles of Distributed Systems, 17-19 December 2018, Hong Kong*. LIPICS, 2018.

[12] J. Czyzowicz, R. Killick, E. Kranakis, D. Krizanc, and O. Morales Ponce. Gathering in the plane of location-aware robots in the presence of spies. In *SIROCCO 2018, Ma'ale HaHamisha, Israel, June 18-21, 2018, Revised Selected Papers*, pages 361–376, 2018.

[13] J. Czyzowicz, A. Kosowski, and A. Pelc. How to meet when you forget: Log-space rendezvous in arbitrary graphs. *Distributed Computing*, 25:165–178, 2012.

[14] J. Czyzowicz, A. Pelc, and A. Labourel. How to meet asynchronously (almost) everywhere. *ACM Transactions on Algorithms (TALG)*, 8(4):37:1–37:14, 2012.

[15] Y. Dieudonné and A. Pelc. Anonymous meeting in networks. *Algorithmica*, 74:908–946, 2016.

[16] Y. Dieudonné, A. Pelc, and V. Villain. How to meet asynchronously at polynomial cost. *SIAM J. Comput.*, 44(3):844–867, 2015.

[17] O. Feinerman, A. Korman, S. Kutten, and Y. Rodeh. Fast rendezvous on a cycle by agents with different speeds. *Theoretical Computer Science*, 688:77–85, 2017.

[18] A. Hoorfar and M. Hassani. Inequalities on the Lambert W function and hyperpower function. *JIPAM. Journal of Inequalities in Pure & Applied Mathematics [electronic only]*, 9, 01 2008.

[19] T. Izumi, Y. Katayama, N. Inuzuka, and K. Wada. Gathering autonomous mobile robots with dynamic compasses: An optimal result. In *International Symposium on Distributed Computing*, pages 298–312. Springer, 2007.

[20] T. Izumi, S. Souissi, Y. Katayama, N. Inuzuka, X. Défago, K. Wada, and M. Yamashita. The gathering problem for two oblivious robots with unreliable compasses. *SIAM J. Comput.*, 41(1):26–46, 2012.

[21] Y. Katayama, Y. Tomida, H. Imazu, N. Inuzuka, and K. Wada. Dynamic compass models and gathering algorithms for autonomous mobile robots. In *International Colloquium on Structural Information and Communication Complexity*, pages 274–288. Springer, 2007.

[22] P. King and F. M. auf der Heide. Continuous protocols for swarm robotics. In *Proceedings of Moving and Computing, 5-9 June, 2017. La Maddalena, Italy*. Springer, 2017.

[23] E. Kranakis, D. Krizanc, and E. Markou. The mobile agent rendezvous problem in the ring. *Synthesis Lectures on Distributed Computing Theory*, 1(1):1–122, 2010.

[24] S. Li, C. Markarian, F. M. auf der Heide, and P. Podlipyan. A continuous strategy for collisionless gathering. In *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*, pages 182–197. Springer, 2017.

[25] A. Pelc. Reaching a target in the plane with no information. *Information Processing Letters*, 140:13–17, 2018.

[26] A. Pelc. *Deterministic Rendezvous Algorithms*. In P. Flocchini, G. Prencipe, and N. Santoro, editors, *Distributed Computing by Mobile Entities*, pages 423–454. Springer, 2019.

[27] A. Ta-Shma and U. Zwick. Deterministic rendezvous, treasure hunts, and strongly universal exploration sequences. *ACM Transactions on Algorithms (TALG)*, 10(3):12, 2014.