Distributed Computing
https://doi.org/10.1007/s00446-024-00463-7

On the power of bounded asynchrony: convergence by autonomous robots with limited visibility

David Kirkpatrick[1] · Irina Kostitsyna[2] · Alfredo Navarra[3] · Giuseppe Prencipe[4] · Nicola Santoro[5]

Received: 2 May 2023 / Accepted: 19 February 2024

Abstract
A distributed algorithm $\mathcal{A}$ solves the POINT CONVERGENCE task if an arbitrarily large collection of entities, starting in an arbitrary configuration, move under the control of $\mathcal{A}$ to eventually form and thereafter maintain configurations in which the separation between all entities is arbitrarily small. This fundamental task in the standard $\mathcal{OBLOT}$ model of autonomous mobile entities has been previously studied in a variety of settings, including full visibility, exact measurements (including distances and angles), and synchronous activation of entities. Our study concerns the minimal assumptions under which entities, moving asynchronously with limited and unknown visibility range and subject to limited imprecision in measurements, can be guaranteed to converge in this way. We present an algorithm operating under these constraints that solves POINT CONVERGENCE, for entities moving in two or three dimensional space, with any bounded degree of asynchrony. We also prove that under similar realistic constraints, but unbounded asynchrony, POINT CONVERGENCE in the plane is not possible in general, contingent on the natural assumption that algorithms maintain the (visible) connectivity among entities present in the initial configuration. This variant, that we call COHESIVE CONVERGENCE, serves to distinguish the power of bounded and unbounded asynchrony in the control of autonomous mobile entities, settling a long-standing question whether in the Euclidean plane synchronously scheduled entities are more powerful than asynchronously scheduled entities.

Keywords Mobile robots · Distributed geometric algorithms · Asynchronous computation

1 Introduction

1.1 Framework and research question

The notion of *distributed computational systems* was initially restricted to systems of processing units interacting by exchanging messages through a fixed communication network. It has broadened over time to include not only other forms of communication (e.g., via shared memory), but also very different types of computational entities and operational environments, from mobile agents to programmable particles, from mobile wireless ad-hoc networks to robot swarms.

The main theoretical questions in all these environments are the same: what are the minimal operational conditions that allow the entities to solve certain fundamental problems?

Some of the results of this paper were presented in preliminary form in [1].

✉ Giuseppe Prencipe
giuseppe.prencipe@unipi.it

David Kirkpatrick
kirk@cs.ubc.ca

Irina Kostitsyna
i.kostitsyna@tue.nl

Alfredo Navarra
alfredo.navarra@unipg.it

Nicola Santoro
santoro@scs.carleton.ca

[1] Department of Computer Science, University of British Columbia, Vancouver, Canada

[2] Department of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands

[3] Department of Mathematics and Computer Science, Università degli Studi di Perugia, Perugia, Italy

[4] Department of Computer Science, Università degli Studi di Pisa, Pisa, Italy

[5] School of Computer Science, Carleton University, Ottawa, Canada

Published online: 09 April 2024

What is the computational relationship between different conditions?

In this paper, we address these questions for systems of autonomous mobile entities, which we refer to as *robots*, within the standard $\mathcal{OBLOT}$ model (see, e.g. [2] and references therein). We focus on one fundamental task, called *convergence*, that asks for robots to congregate in a region of arbitrarily small diameter.

In the $\mathcal{OBLOT}$ model, identical robots are free to move in a featureless spatial universe, typically the two dimensional Euclidean space. Very few papers deal with higher dimensions like the three dimensional space approached in [3–6].

Robots operate under restrictions designed to elucidate what is essential in solving certain robot coordination objectives. Specifically, robots are viewed as points, and communicate only indirectly through their position which is visible to other robots within some, possibly bounded and unknown, visibility range. Robot motion is determined locally by a resident algorithm that depends only on current perceptions of the location of other robots, relative to some private coordinate framework. The activation of robots and realization of intended motions is governed by an adversarial, but fair, scheduler.

Computations in the $\mathcal{OBLOT}$ model are very different from those in other distributed systems (e.g., message-passing networks): they are just continuous re-positionings of robots, realized by loosely coordinated activity of the individual robots. Computations start from some (typically unconstrained) initial configuration, and move towards some goal configuration. Permissible goal and intermediate configurations are expressed in terms of temporal geometric predicates.

A computational task simply specifies a subset of computations that *fulfill* the task, and an algorithm is said to *solve* the task if, under its control, starting from any permissible initial configuration, the robots produce a computation that fulfills the task *in all valid activation sequences*. The validity of activation sequences is determined by the level of *synchronization* of the system; as in other distributed environments, the primary distinction is between synchronous and asynchronous scheduling models.

In the *semi-synchronous* model (SSYNC), time is divided into discrete intervals, called *rounds*; in each round, the scheduler specifies a subset of the robots that perform some elementary activity in perfect synchronization. (The only constraint is *activation fairness*: every robot is activated within finite time and infinitely often.) When all robots are activated in every round, the model is said to be *fully-synchronous*.

In contrast, in the *asynchronous* model (ASYNC), robots are activated at arbitrary times (constrained again by activation fairness), independently of the other robots, and the duration of individual activities, though finite, is unpredictable.

There are many studies on the feasibility of tasks and the complexity of algorithms in both the synchronous and asynchronous settings; Sect. 2 provides an overview of the most relevant related work. In spite of long and intensive research efforts, a fundamental question has remained unsettled: *Is there any difference in computational scope between algorithms working within the scheduling models* SSYNC *and* ASYNC*?* More precisely: Does the assumption of synchronous scheduling of activity make possible the solution of some task that is not solvable in the asynchronous setting?

In this paper, we provide an affirmative answer to the question. In fact, our results, which apply to realistic robots that operate with imprecise sensing, demonstrate an even stronger separation: between scheduling models with unbounded asynchrony (ASYNC) and bounded asynchrony (what we call k-ASYNC). In the latter the degree of asynchrony is bounded by an arbitrarily large but fixed k, that is, while any one robot is active, any other robot can be activated at most k times.

1.2 Robot gathering and convergence

Most of the research concerning the limits of computation within the $\mathcal{OBLOT}$ model (see Sect. 2 for a more comprehensive overview) has concentrated on specific tasks that fit under the general framework of *pattern formation* [5, 7–11].[1] Among these, the task of *point formation*, where all robots are required to congregate (and remain) at the same point (not fixed in advance), has attracted particular attention due to its correspondence with the basic coordination task in swarm robotics called GATHERING (or RENDEZVOUS) [4, 12–20]. The less stringent version of this task, called POINT CONVERGENCE (hereafter simply CONVERGENCE) [21–30], requires the robots to move in such a way that, for all $\varepsilon > 0$, a configuration in which the separation between all robots is at most ε is eventually reached and maintained.

Previous investigations have examined GATHERING and CONVERGENCE, taking into account a variety of different factors (e.g., synchronicity, shared knowledge of the environment, and inaccuracies in measurement and motion) assuming *unlimited visibility*. In brief, existing algorithms have been shown to solve GATHERING assuming one or more of the following: (i) fully synchronous (FSYNC) scheduling, (ii) shared knowledge of a reference coordinate system, (iii) absence of co-located robots in the initial configuration, or (iv) the use of non-computable geometric predicates. Under even slightly relaxed (SSYNC) scheduling assumptions, GATHERING is known to be impossible in general,

[1] Many other tasks can be reduced to pattern formation ones (e.g., leader election, gathering) or require pattern formation as an integral step (e.g., flocking).

whereas CONVERGENCE is known to be solvable, provided multiplicities (i.e. the co-location of robots) can be detected, even with fully asynchronous (ASYNC) scheduling.

Comparatively few investigations have focused on the *limited visibility* [1, 21, 27, 31–34]. It is typically assumed that the visibility graph associated with the initial robot configuration is connected, and that the number of robots is not known, even approximately, obviously not an issue in the full visibility setting.

The design of successful algorithms for robots with limited visibility in the ASYNC scheduling model, has come with the assumption of additional robot capabilities. For instance GATHERING has been shown to be solvable if the local coordinate systems of all robots agree on the direction of the axes.

In the SSYNC scheduling model, the pioneering work of Ando et al. [21] (described in detail in Sect. 4.1) established the feasibility of CONVERGENCE, under limited visibility.

In subsequent work [27], Katreniak introduced the k-ASYNC scheduling model.[2], that permits bounded asynchrony in robots' activation, and presented a solution algorithm for CONVERGENCE in the 1-ASYNC model (a solution under a more restricted version of the 1-ASYNC scheduler was presented earlier in [19]). The existence of an algorithm to solve CONVERGENCE in the k-ASYNC model, for $k > 1$, was left as an open question.

It is worth pointing out that, like all known algorithms that deal with limited visibility, the algorithms of Ando et al. and Katreniak have the property that, if two robots are initially within visibility range of each other, they remain mutually visible thereafter. Thus they solve an apparently more restricted variant of CONVERGENCE, that we call COHESIVE CONVERGENCE.

1.3 Main contributions

The focus of this paper, as in those of Ando et al. [21] and Katreniak [27], concerns the COHESIVE CONVERGENCE task for identical autonomous robots with bounded visibility (where the visibility radius is possibly unknown to the robots). This requires any collection of such robots, starting from an arbitrary connected configuration, to move, under the control of a possibly adversarial scheduler, in such a way that, for all $\varepsilon > 0$, the robots are guaranteed to reach and maintain a configuration in which all robots lie within a circle of radius at most ε. Furthermore, all robot pairs that are mutually visible in the initial configuration remain so thereafter.

We prove that, considering algorithms that tolerate a very modest amount of measurement imprecision, COHESIVE-CONVERGENCE is *solvable* in the k-ASYNC scheduling model, for any fixed k, providing a strong positive answer to a question left open in [27]. However, even under slightly weaker assumptions, the same task is shown to be *unsolvable* in the ASYNC scheduling model. In this sense, COHESIVE CONVERGENCE provides a separation between scheduling models with bounded and unbounded asynchrony, and ipso facto between the SSYNC and ASYNC scheduling models.

In particular, our primary contributions are threefold: (i) a novel moderately error-tolerant algorithm that succeeds in the scheduling model k-ASYNC, for any fixed k, an environment that permits scheduling of robot activity with an arbitrarily large but bounded degree of asynchrony, (ii) an analysis that allows us to go well beyond what is possible with previous techniques that depend on the assumption of a significantly more restricted scheduling environment (SSYNC or 1-ASYNC), shedding new light on the power of bounded asynchrony, and (iii) a family of robot configurations that demonstrates the impossibility of solving COHESIVE CONVERGENCE in the fully asynchronous scheduling environment ASYNC, provided algorithms are required to tolerate a modest amount of imprecision in perception (yet significantly less than what can be tolerated by our k-ASYNC algorithm). Furthermore, our construction shows the impossibility of even weaker forms of COHESIVE CONVERGENCE, where visibility edges are allowed to be broken as long as the whole robot configuration remains connected. Our algorithm is described in its basic form in Sect. 4, with extensions, including error-tolerance, and three-dimensional generalization, outlined in Sect. 7. Even in the same scheduling environment, our algorithm is slightly simpler in formulation and provides a more general solution than its predecessors developed in [19, 21, 27]. In particular, the intended destination point for a robot in each step depends only on the directions to the pair of neighbours, among those that are relatively distant, that define the maximum angular range. Furthermore, distance and angle measurements to neighbours are assumed to be accurate only to within some limited imprecision, and robot motion is subject to some limited error in both distance and angle. Finally, our algorithm exhibits a degree of uniformity not evident in its predecessors: specifically, (a) dependence on the visibility range V is not built into the algorithm, and (b) our algorithm is simply formulated to work in an environment with a base level of asynchrony; to function in an environment that permits a higher degree of asynchrony, captured by a parameter k, one can simply scale the motion function used by our algorithm in its base formulation by a factor α that is at most $1/k$.

As with the analysis of algorithms in [19, 21, 27], our proof of cohesive convergence involves (i) an argument that the edges of the initial visibility graph are preserved, and (ii) an argument that congregation, i.e., convergence to an arbitrarily small region, eventually takes place. Our analysis, presented in Sect. 5, is complicated by the fact that, harnessed

[2] In his paper, Katreniak named the model *k-bound asynchrony* We shall call it k-ASYNC, to conform to the naming convention of the other models.

by an adversarial scheduler, even the most basic level of asynchrony makes it impossible for a robot $\mathbb{X}$ to accurately discern the influence of its own location on the motion of an active neighbour $\mathbb{Y}$. This is further compounded by the fact that, in environments with a higher degree of asynchrony, neighbour $\mathbb{Y}$ might have seen $\mathbb{X}$ many times during its current motion. In order to help isolate the most salient features of our analysis in asynchronous environments, we first introduce a restriction on the k-ASYNC environment, requiring intersecting activity intervals to nest, one entirely within the other. By staging our analysis in this way, we provide a clearer picture of the extent to which the difficulties associated with asynchrony in our setting arise from (possibly lengthy) chained, as opposed to nested, activations. (The reader might find a glance at Fig. 2 helpful in distinguishing between these models.) To demonstrate the preservation of visibility under arbitrarily long chains of activations we introduce a novel backward reachability analysis that fully exploits the relative simplicity of our algorithm's motion function.

Our congregation argument is presented in Sect. 6. We start by observing, as was the case in [19, 21, 27], that convergence to a convex configuration with bounded diameter follows directly from the fact that the convex hull of successive configurations are properly nested. Our argument that the limiting diameter is zero differs from, and is arguably simpler than, that used in these earlier works. The argument exploits a kind of hereditary property that ensures that, after some point in time, robot locations become relatively stable in the sense that once a robot has vacated a small neighbourhood of an extreme point of the configuration it must remain outside of that neighbourhood.

Demonstrating the impossibility of solving COHESIVE CONVERGENCE in any environment that assumes exact measurement and control seems daunting, since, assuming motion is rigid, such assumptions might open up the possibility of encoding information, perhaps history, in the locations of robots. However, in Sect. 8, we show that algorithms that control more realistic robots (operating with even a very modest amount of imprecision in measurements) cannot solve COHESIVE CONVERGENCE in a scheduling environment with no bound on asynchrony (other than a fairness condition that ensures that no robot is permanently locked out of activity). This is demonstrated by a family of configurations with the property that for any algorithm there is a configuration in the family that can be forced to become disconnected into two linearly separable components. Interestingly, the adversarial scheduler that achieves this uses only nested activations (of necessarily unbounded depth).

Our work raises several interesting questions, suitable for future research. Several of these are highlighted in our concluding section (Sect. 9).

1.4 Outline

The paper is organized as follows. In the next section, we review some relevant literature on POINT CONVERGENCE. Section 3 specifies the model under which we approach the problem. Section 4 contains the description of our new algorithm. Section 5 is devoted to the analysis of the proposed algorithm concerning visibility properties. Section 6, instead, is devoted to prove the congregation property of the algorithm. Section 7 discusses some possible generalizations of the proposed algorithm in terms of error-tolerance and three-dimensional environment. Section 8 is devoted to prove the impossibility result for asynchronous robots. Finally, Sect. 9 provides conclusive remarks and challenging research directions for future work.

2 Related theoretical work

As mentioned in the introduction, most of the existing research addressing the relative strength of different scheduling models (in particular SSYNC and ASYNC) has concentrated on the study of the tasks of forming or converging to geometric patterns.

2.1 Pattern formation

Almost all studies on pattern formation have assumed *unlimited visibility*, and that measurements and computations are free of error. The first question studied has been what patterns can be formed from a given initial configuration [35]. Assuming that robots share a common handedness (also called *chirality*), a sequence of results has shown that the set of formable patterns depends solely on the degree of symmetry (a parameter called *symmetricity*) of the initial configuration ([9, 35, 36]; see also [37]). In other words, with chirality, there is no distinction between the patterns formable from an initial configuration in SSYNC and ASYNC. No similar general results are known without chirality.

The related question of determining from which initial configurations it is possible to form any possible pattern, called ARBITRARY PATTERN FORMATION, was posed and studied [10] in ASYNC for robots without chirality; the answer to this question has been provided in [8], where it has been shown that the set of possible initial configurations coincides with those from which it is possible to solve another basic task, namely LEADER ELECTION.

For the converse question of which patterns can be formed from every initial configuration, there are only two possible candidate geometric patterns: *point* and *uniform circle*, and the latter only if the robots initially occupy distinct locations. The two tasks, POINT FORMATION and UNIFORM CIRCLE FORMATION, have been intensively studied assuming that the

robots initially occupy distinct locations. Remarkably, both tasks have been shown to be solvable in ASYNC even without chirality and with non-rigid movements[3] [15, 38], assuming multiplicity detection[4] in the case of POINT FORMATION.

Observe that POINT FORMATION corresponds to a basic coordination task in swarm robotics, called GATHERING or RENDEZVOUS, where all robots are required to meet at a common point (not fixed in advance). It is known that this task, without agreement on the coordinate system, is *unsolvable* for $n = 2$ robots even in SSYNC [35]; expanding on this result, it has been shown that, without any additional capability of the robots (e.g., global coordinate system, multiplicity detection), it is unsolvable in SSYNC also for $n > 2$ [20]. Indeed POINT FORMATION had been conjectured to be unsolvable in ASYNC even with multiplicity detection and initially scattered robots (i.e., at distinct initial locations).

These basic impossibilities have further motivated the investigation of a weaker version of POINT FORMATION, called POINT CONVERGENCE (or simply CONVERGENCE) discussed in the next section, which requires the robots to move in such a way that, for all $\varepsilon > 0$, a configuration in which the separation between all robots is at most ε is eventually reached and maintained.

The presence of even limited inaccuracy of the measurement of both distances and angles of the robots renders POINT FORMATION impossible in SSYNC even with a global coordinate system and strong multiplicity detection [26]. On the other hand, POINT FORMATION has been shown to be solvable with chirality in SSYNC even in presence of a limited tilt (less than $\pi/4$) of the local compasses [18]. Further, with agreement on one axis, POINT FORMATION is solvable in ASYNC even in presence of *occlusions* (a robot $\mathbb{X}$ looking at a robot $\mathbb{Y}$ cannot see anything beyond $\mathbb{Y}$ along the ray from $\mathbb{X}$ that passes through $\mathbb{Y}$) and crash faults [39].

Related investigations on pattern formation with unlimited visibility include the study of: the impact of crash and Byzantine faults among the robots (e.g., [12, 13, 26, 39, 40]); the EMBEDDED PATTERN FORMATION task, where the pattern is given to the robots as visible points in the plane, solved in ASYNC by robots without chirality [41]; and the PATTERN SEQUENCE task, where robots are required to form an ordered sequence of patterns, shown to be solvable in SSYNC by robots with chirality [42].

Very few investigations have focused on pattern formation in the *limited visibility* setting. In this setting, it is assumed that the visibility graph associated with the initial robot configuration is connected. It is has been shown that limited visibility drastically reduces the set of patterns that can be formed by robots in ASYNC [34]. On the other hand, POINT FORMATION is solvable in ASYNC from any initial configurations if the robots are endowed with global consistency of the local coordinate system [32]. We are not aware of any investigation on pattern formation in the limited visibility setting that considers measurement inaccuracies.

2.2 Point convergence

The basic impossibilities of POINT FORMATION have further motivated the investigation of a weaker version of the task called POINT CONVERGENCE (or simply CONVERGENCE), which requires the robots to move in such a way that, for all $\varepsilon > 0$, a configuration in which the separation between all robots is at most ε is eventually reached and maintained. Observe that, by definition, any solution to POINT FORMATION under some assumptions automatically solves POINT CONVERGENCE under the same assumptions.

2.2.1 Point Convergence with unlimited visibility

The result of [15] on POINT FORMATION implies that, with multiplicity detection, POINT CONVERGENCE is solvable in ASYNC from any initial configuration where the robots occupy distinct locations.

If the multiplicity detection is strong, POINT CONVERGENCE is solvable in ASYNC from *any* initial configuration by means of the simple `Centre_Of_Gravity` (`CoG`) algorithm, where robots move toward the center of gravity (a.k.a. center of mass, baricenter) of the configuration [25]; note that the center of gravity may change as robots move. The convergence rate of the `Centre_Of_Gravity` algorithm was subsequently improved using the `Center_Of_Minbox` (`GCM`) strategy, where the *minbox* is the minimal axes aligned box containing all current robots' positions [24].

These feasibility results have led to the investigation of POINT CONVERGENCE under stronger adversarial conditions, possibly with some additional robots capabilities. In particular, studies have considered possible inaccuracies of measurements, in terms of distances or angles, of the robots. A modification of the `CoG` algorithm, called `Restricted_CoG`, has been shown [26] to allow the robots to converge in SSYNC even if distances are measured only within an multiplicative accuracy parameter $\epsilon < 1/2$ known to the robots; interestingly, in $\mathbb{R}^1$ (i.e., on the line), this algorithm would allow convergence in ASYNC. If the imprecise measurements are uniform, i.e., there is a uniform error ratio for all the distances and angles perceived during an observation by a robot, then POINT CONVERGENCE can be solved in a significantly wider set of cases [30].

[3] i.e., when active, a robot is not guaranteed to reach its computed destination unless the distance toward the target is smaller than an unknown but fixed threshold.

[4] i.e., ability to detect whether more than one robot is at the same location; it is said to be *strong* if the exact number of robots at the same location can be detected.

The case has also been studied where robots cannot measure the distance to other robots, but only detect their *proximity* (i.e., whether or not another robot is closer than a given constant distance δ) [29].

2.2.2 Point Convergence with limited visibility

The pioneering work on POINT CONVERGENCE under limited visibility is by Ando et al. [21], in the SSYNC scheduling model. The authors propose and analyze an algorithm that leads to convergence, by means of *cautious* moves, starting from any connected configuration of robots. In their `Centre_Of_SEC` algorithm, each active robot takes a snapshot of its neighbourhood (up to a known limited visibility range V), computes the center of the *Smallest Enclosing Circle* (SEC) of the perceived robots,[5] and then makes a move toward the center of the SEC for a distance that guarantees to preserve the visibility of these perceived robots, regardless of the movements of other robots following the same protocol. (Quite recently, Braun et al. [3] considered an extension of the `Centre_Of_SEC` algorithm, applied in the three dimensional Euclidean space, assuming a continuous time variant of the scheduling model FSYNC.)

In subsequent work [27], Katreniak introduced the more powerful scheduling model k-ASYNC, that permits robot activations with bounded asynchrony, and presents a solution algorithm for POINT CONVERGENCE in the model 1-ASYNC (a solution under a more restricted version of the 1-ASYNC scheduler was presented earlier in [19]). The resolution of CONVERGENCE in the model k-ASYNC, for $k > 1$, was left as an open problem.

A variant of POINT CONVERGENCE is the NEAR- GATHERING task, that requires the robots to converge without two (or more) robots ever occupying the same point at the same time. In [43], it is shown that NEAR- GATHERING can be solved with limited visibility in the ASYNC model when all the robots are assumed to have a compass (hence they agree on the "North" direction), but they do not necessarily have the same handedness (hence they may disagree on the "West").

It is worth pointing out here that these, like all known POINT CONVERGENCE algorithms that deal with limited visibility, have the property that if two robots are initially within visibility range of each other, they remain so in all subsequent configurations, thereby guaranteeing COHESIVE CONVERGENCE.

[5] The smallest circle that encloses all the robots perceived during the `Look` phase at distance not greater than V.

3 Model

We consider the standard $\mathcal{OBLOT}$ model of distributed systems of mobile entities (e.g., see [44]). The system is composed of a set $\mathcal{R} = \{\mathbb{X}^1, \ldots, \mathbb{X}^n\}$ of $n \geq 1$ autonomous computational entities, called *robots*, that reside in d-dimensional Euclidean space $\mathbb{R}^d$, $d \geq 1$, within which they operate in discrete `Look-Compute-Move` activity cycles.

3.1 Robots

We treat robots as dimensionless entities (points), free to move in any direction within $\mathbb{R}^2$; a discussion of extensions of our results to motion in $\mathbb{R}^3$ is left to Sect. 7. We use X, sometimes subscripted, to denote the generic location of robot $\mathbb{X}$; when needed $X(t)$ is used to denote the location of $\mathbb{X}$ at a specific time t. The multiset $\mathcal{C}(t) = \{X(t) : X \in \mathcal{R}\}$ specifying the locations of the robots at time t is called the *configuration* of the system at time t.

Robots are equipped with sensors that allow them to observe the positions of the other robots within a fixed, but possibly unknown range V. More precisely, the *visible region*[6] of robot $\mathbb{X}$ at time t is the closed point set[7] $Vis(\mathbb{X}, t) = \{P : |X(t)P| \leq V\}$. We say that a robot $\mathbb{Y}$ is a *neighbour* of robot $\mathbb{X}$ at time t if $Y(t)$ belongs to $Vis(\mathbb{X}, t)$. The associated *visibility graph* at time t is the undirected graph $G(t) = (\mathcal{R}, E(t))$, where $(\mathbb{X}, \mathbb{Y}) \in E(t)$ if and only if $|X(t)Y(t)| \leq V$. A configuration of robots is said to be *connected* when its associated visibility graph is connected.

Robots are provided with a persistent read-only memory containing a control algorithm as well as a local volatile memory for its computations, that are assumed to be error-free.

Robots are (i) *autonomous*: that is, they operate without a central control or external supervision; (ii) *identical*: that is, they are indistinguishable by their appearance, they do not have distinct identities that can be used during the computation, and they all have and execute the same algorithm; and (iii) *silent*: they have no means of direct communication of information to other robots, so any communication occurs in a totally implicit manner, by moving and by observing the positions of the robots within visibility range.

3.2 Activity cycles and obliviousness

The behaviour of each robot can be described as the alternation of finite length *activity* intervals with finite length *inactivity* intervals. Each activity interval consists of a sequence of three phases: `Look`, `Compute`, and `Move`,

[6] We will argue in Sect. 7 that a sharp visibility threshold is not essential for the correctness of our algorithm.

[7] If P and Q are points in $\mathbb{R}^d$, we denote by $\overline{PQ}$ the line segment joining P and Q, and by $|PQ|$ the length of this segment.

called a Look-Compute-Move (LCM) activity cycle. The operations performed by each robot $\mathbb{X}$ in each phase are as follows.

1. Look At the start of an activity interval, the robot observes the world within its visibility range. The result is an instantaneous *snapshot* of the positions[8] occupied by visible robots at that time; these positions are expressed within a local (i.e., private) coordinate system. The private coordinate systems of different robots at the same time, or the same robot at different activation cycles, need not be consistent; hence from a global point of view, the robots are *disoriented*.
2. Compute Using the snapshot as an input, the robot executes its built-in algorithm $\mathcal{A}$, the same for all robots, which determines an intended *destination* point in the local coordinate system of the robot.
3. Move The robot moves toward the computed destination along a straight trajectory. This phase (and with it, the full activity cycle) might end before the robot reaches its destination.

The duration of each Compute and Move phase is assumed to be finite,[9] while the Look phase is assumed to be instantaneous. During a Move phase a robot is said to be *motile* (not necessarily moving, but capable of moving), otherwise it is *immotile*.

Since the local memory of robots is volatile, it cannot be assumed that memory contents persist between activity cycles. Accordingly, the control algorithm is *oblivious*: the Compute step is independent of past actions and computations.

3.3 Adversarial control and conditions

In the environment in which the robots operate, there are several factors and conditions, in addition to the initial configuration, that are beyond the control of their algorithm but nevertheless impact their actions and, ultimately, their worst-case behaviour. These factors, which are assumed to be under adversarial control, are discussed in the following.

3.3.1 Activation and synchronization

The mapping of robot activities to time, including activation/deactivation times, as well as the duration of Compute and Move phases within each activity cycle, and the rate of motion within each Move phase, is determined by a *scheduler* that is constrained only by the understood level (or model) of system synchronization. In all models, the scheduler must respect *activation fairness*: for each robot $\mathbb{X}$ and each time t, there exists a time $t' > t$ where $\mathbb{X}$ will be active.

The main models are the synchronous and the asynchronous ones (refer to Fig. 1):

- In the *synchronous* (also called *semi-synchronous*) model (SSYNC) time is logically divided into a sequence of *rounds*. In each round, whose duration is not necessarily fixed, a *subset* of the robots is activated, and they perform each phase of their LCM cycle simultaneously,[10] terminating their activity interval by the end of the round. The choice of which robots are activated in a round is made by the scheduler. A special case of SSYNC is the *fully-synchronous* model (FSYNC) where all robots are activated in every round.
- In the *asynchronous* model (ASYNC), each robot is activated at arbitrary times, independently of the others. In each activity interval, the duration of each Compute and Move phase is finite but unpredictable, and might be different in different cycles. The duration of the Compute and Move phases of each cycle as well as the decision of when to activate a robot is controlled by the scheduler.

The asynchronous model licenses an adversarial scheduler to depart from the synchronous model in two ways: (i) activity intervals of different robots can overlap arbitrarily; and (ii) one robot can be activated arbitrarily many times during a single activation of another robot. To better understand the impact of these as separate factors, we first define the *nested-activation* model (NESTA) that restricts the activity intervals of all pairs of robots to be either disjoint or nested (no activity interval contains exactly one endpoint of another activity interval). We will focus more precisely on two natural restrictions of the models k-NESTA and k-ASYNC, that lie between the models SSYNC and ASYNC (see Fig. 2):

- *k-Nested-Activations* (k-NESTA) Activity intervals of any pair of robots are either disjoint or nested, and in addition, at most k activity intervals of one robot can be nested within a single activity interval of another.
- *k-Asynchronous* (k-ASYNC) As in ASYNC, robots are activated independently and the duration of each activity interval is arbitrary (but finite). However, at most k activations (i.e., the starts of the activation intervals) of one robot can occur within a single activity interval of another.

[8] If the robots are endowed with multiplicity detection, the snapshot indicates also whether there are two or more robots co-located at a point; otherwise such multiplicities are perceived as a single robot.

[9] Note that also the Compute phase could be assumed to be instantaneous, without any loss of generality.

[10] Hence the duration of each Compute and Move phase can be thought to last always the same, for all activity intervals.

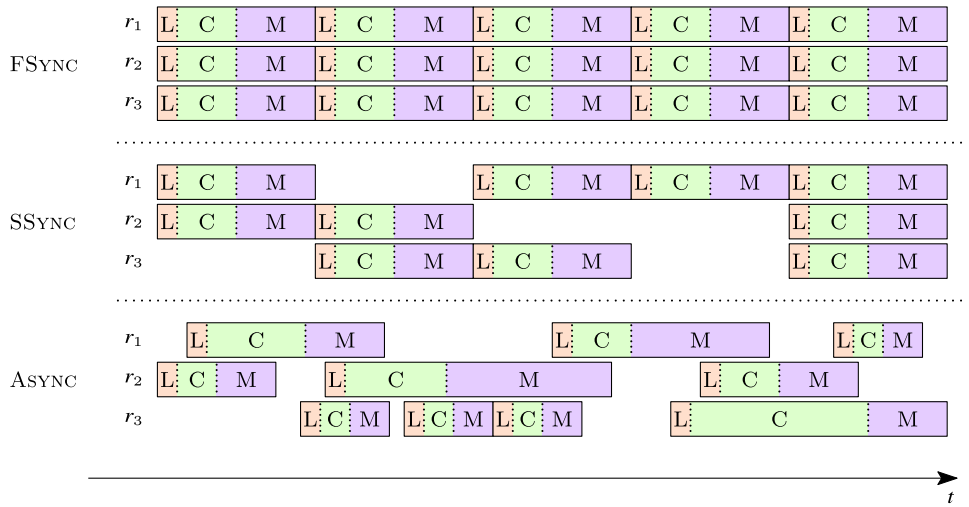

Fig. 1 Illustration of activity intervals in the FSYNC, SSYNC, and ASYNC scheduling models, emphasizing their differences

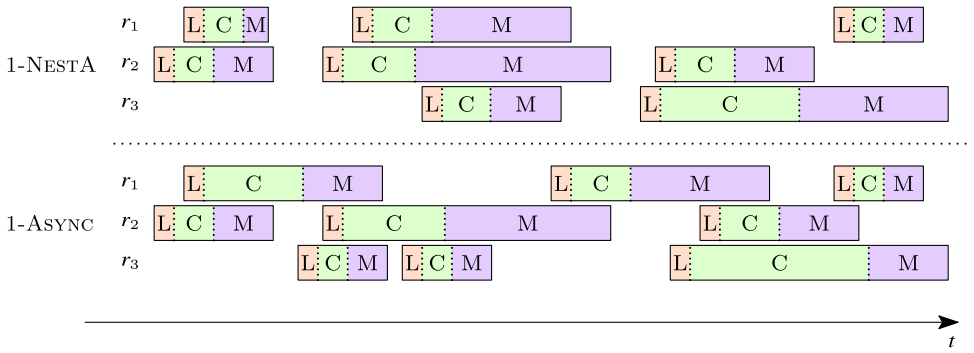

Fig. 2 Illustration of activity intervals in the 1-NESTA and 1-ASYNC scheduling models

3.3.2 Rigidity of motion

In the `Move` phase, there is a constant $\xi \in (0, 1]$ (unknown to the robots, and possibly related to the visibility range V) such that robots are guaranteed to move at least fraction ξ of the way toward their intended destination. The associated motion is said to be ξ-*rigid*. (If $\xi = 1$, motion is simply said to be *rigid*). Actually, in the literature, non-rigid motion is usually qualified by an assumption that at least some non-trivial distance is traversed. In our setting, where it makes no sense for an algorithm to specify a motion of length that significantly exceeds the visibility range, this assumption can be subsumed by the ξ-rigid assumption, for suitable ξ. The correctness of our algorithm depends only on the assumption that motion is ξ-rigid for some (possibly unknown) $\xi > 0$.

It is also possible for the actual destination to deviate from the trajectory to the intended destination. However, if this deviation could exceed some absolute constant, then POINT CONVERGENCE is clearly not possible, even in the fully synchronous model. Our algorithm is formulated with the assumption that there is no trajectory deviation, but can be modified to work with a deviation that is bounded by some quadratic function of the motion length.

3.3.3 Measurement imprecision

In error-tolerant settings, the accuracy of the measurements (distances and angles) made by a robot during its `Compute` phase, as well as that of a robot's ability to realize its intended trajectory during its `Move` phase, are also considered to be subject to adversarial control.

However, it should be clear that if some absolute error is possible in distance measurements, then POINT CONVERGENCE is impossible, even with a benevolent scheduler. Furthermore, if some absolute error in angle measurements is possible then again POINT CONVERGENCE is impossible, even with a benevolent scheduler. This follows by observing that the presence of such error would make it impossible to distinguish some nearly co-linear triple of robots, separated by distance V, with a co-linear such triple. Given this, any algorithm would be forced to refrain from moving the middle of such a co-linear triple (at the risk of a catastrophic separation). This in turn would lead to completely frozen

activity on a suitably large collection of robots configured as a regular polygon with vertex separation V. Our algorithm can be modified to succeed in the presence of a modest amount of relative error in both distance and angle measurements, provided the latter arises in a way that does not allow a non-co-linear triple of robots to be confused with a nearby co-linear triple.

3.4 Tasks and solutions

Since robots in the $\mathcal{OBLOT}$ model can only observe the positions of others and then move, a task to be performed is expressed in terms of a *temporal geometric predicate*. From some point in time onward, the configurations formed by the robots' positions must satisfy this predicate.

Let $\mathcal{R}$ be an arbitrary collection of robots. The POINT CONVERGENCE task requires the robots in $\mathcal{R}$, starting in an arbitrary connected configuration[11] where they are all inactive, to become arbitrarily close. More precisely, POINT CONVERGENCE is the task defined by the temporal geometric predicate:

$$\texttt{Convergence} \equiv \forall \varepsilon \in \mathbb{R}^+, \exists t : \forall t' \geq t, \forall \mathbb{X}, \mathbb{Y} \in \mathcal{R}, |X(t')Y(t')| \leq \varepsilon.$$

A more constrained version is the COHESIVE CONVERGENCE task that additionally requires that, at all times, the visibility graph is a supergraph of the initial visibility graph. In fact, it is defined by the temporal geometric predicate:

$$\texttt{CohesiveConvergence} \equiv \texttt{Convergence} \wedge (\forall t \geq 0, E(0) \subseteq E(t)).$$

An algorithm $\mathcal{A}$ is said to *solve* the POINT CONVERGENCE (or COHESIVE CONVERGENCE) task if it satisfies the corresponding predicate, starting from *any* valid initial configuration of robots (of arbitrary size), under *any* (possibly adversarial) scheduler that respects the associated synchronization model. As previously observed, even though COHESIVE CONVERGENCE is strictly more constrained than POINT CONVERGENCE, all known algorithms for POINT CONVERGENCE also solve COHESIVE CONVERGENCE.

4 A new algorithm for COHESIVE CONVERGENCE

This section introduces the KKNPS algorithm, a new algorithm for solving COHESIVE CONVERGENCE that succeeds in the k-ASYNC scheduling model. We begin with a review of the algorithms of [21, 27], emphasizing the features common to the KKNPS algorithm, and features on which the KKNPS algorithm differs. This is followed by a detailed overview of our new approach.

4.1 The COHESIVE CONVERGENCE algorithms of Ando et al. and Katreniak

The algorithm of Ando et al. [21] proceeds as follows: upon activation, each robot $\mathbb{X}$

- locates, according to its local coordinate system, all of the other robots within its visibility range;
- computes a *safe region* for motion with respect to each of its neighbours, and
- computes the center of a minimum enclosing ball of all the robots within its visibility range, and moves as far as possible towards this center while remaining inside all of the individual safe regions.

The algorithm of Ando et al. is formulated in the SSYNC scheduling model. It assumes:

- robots are points that execute their full planned motions instantaneously, i.e., rigid collision-free motion[12];
- knowledge of the common visibility radius V is built into the algorithm[13];
- there is no error in perception or in the realization of planned motion.

The algorithm of Katreniak [27] has many similarities to that of [21]: upon activation, each robot $\mathbb{X}$

- locates, within its local coordinate system, all of the other robots within its visibility range,
- computes a *safe region* for motion with respect to each of its neighbours, and
- moves as far as possible from its current position, while remaining inside a composite safe region that respects all of the individual safe regions.

Katreniak's algorithm is formulated in the 1-ASYNC model. It assumes:

- the common visibility radius V is not known; instead, each robot $\mathbb{Y}$ works, in each `Look` step, with a lower bound $V_{\mathbb{Y}}$ on V, determined by the distance to its furthest neighbour;

[11] It is possible to define Convergence in a meaningful way that applies to unconnected initial configurations as well. This is discussed further in Sect. 7.

[12] The assumption of rigid motion, though standard at the time of [21], could be relaxed without impacting the correctness of the algorithm.

[13] Again, this assumption is stronger than necessary: it would suffice to replace V by the distance to the furthest neighbour.

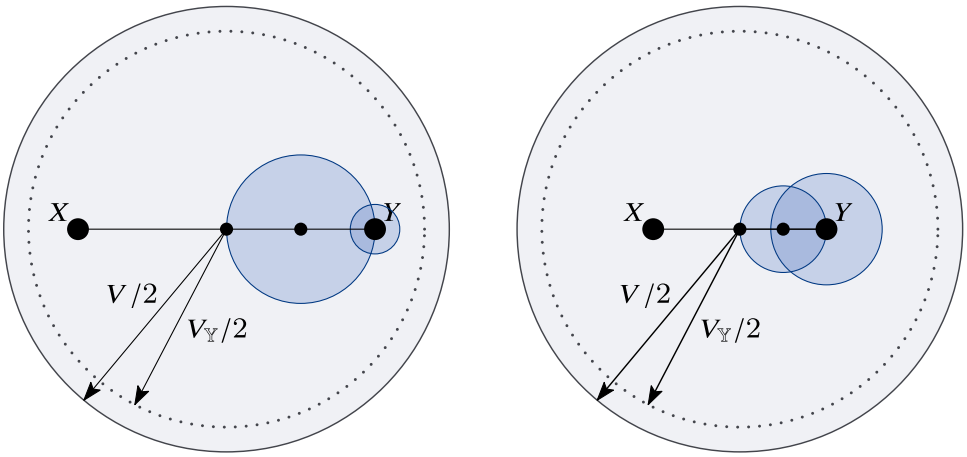

Fig. 3 Basic safe region for motion of robot $\mathbb{Y}$ at location Y (with respect to visible robot $\mathbb{X}$ at location X), as specified by Ando et al. (grey), by Katreniak (blue), when $|XY| > V_{\mathbb{Y}}/2$ (left) and $|XY| \leq V_{\mathbb{Y}}/2$ (right) (colour figure online)

- there is no error in perception;
- the planned motion might not be fully realized, subject to a natural *progress* condition, that ensures eventual convergence.

The most apparent difference between the algorithms of Ando et al. and Katreniak lies in the specification of their safe regions (see Fig. 3). For a robot $\mathbb{Y}$ located at point Y viewing a robot $\mathbb{X}$ located at point X, Ando et al. specify a safe region as a disk with radius $V/2$ centred at the midpoint between X and Y. Katreniak's safe region is formed by the union of two disks, one with radius $|XY|/4$ centred at the point $(X + 3Y)/4$, and the other with radius $(V_{\mathbb{Y}} - |XY|)/4$ centred at Y.

The correctness of the algorithms for Point Convergence in both [21, 27] rest on two observations: (i) *visibility preservation* the choice of safe region guarantees that all robot pairs that start or become mutually visible will remain mutually visible thereafter, and (ii) *incremental congregation* the trajectories of robots following the algorithm exhibit a notion of "progress" towards convergence (measured in terms of the shrinking of the convex hull of the robot locations). In both cases, the argument is complicated by the need to demonstrate that the limit of convergence is a single point.

Note that, even when $k = 1$, both the models k-NestA and k-Async provide modest generalizations of the model SSync. Specifically, by definition, any algorithm that guarantees convergence in the model 1-NestA (or 1-Async) will do the same in the model SSync.

Observation 1 *There exist configurations where the algorithm of Ando et al. fails to maintain a connected visibility graph in both the the 1-*Async *and 2-*NestA *models.*

Proof Figure 4 demonstrates such a configuration with just five robots

In this example, $\mathbb{A}$, $\mathbb{B}$, and $\mathbb{C}$ are stationary throughout (i.e. the scheduler does not activate these robots); and $\mathbb{X}$ and $\mathbb{Y}$ are activated according to the two activation timelines shown on the bottom of the figure. The horizontal axis represents the time, grey arrows designate the snapshots, and the corresponding observed positions of the robots $\mathbb{X}$ and $\mathbb{Y}$ are also shown in grey. The red circle, centered at X_2, is the smallest enclosing circle of the points X_0, Y_0, B, and C. The black circle is centered at X_2 and has radius V. In both cases, robot $\mathbb{X}$ moves from location X_0 to location X_1 and then to X_2, and robot $\mathbb{Y}$ moves from Y_0 to Y_1, leading to the distance between the two robots being strictly greater than V.

(i) In general, the algorithm of Ando et al. (unmodified) does not succeed in the model 1-Async, even if it can be assumed that moves are rigid and instantaneous (so the scheduler cannot pause or stop robots at intermediate points in their planned trajectory)[14]; see the timeline in Fig. 4a;

(ii) The same construction, with a slightly modified timeline (Fig. 4b), shows that the algorithm of Ando et al. (unmodified) does not succeed in the 2-NestA model;

□

Remark Using a similar argument, it is not difficult to show that Katreniak's algorithm (unmodified) does not succeed in the k-Async scheduling model, when k is sufficiently large. This no doubt explains why the extension of Katreniak's 1-Async algorithm (for $k > 1$) was left as an open problem in [27]. The reader might well suspect that the failings of the algorithms of Ando et al. and Katreniak are due to overly

[14] Note that Katreniak makes a similar observation, using an example that exploits non-rigid motion.

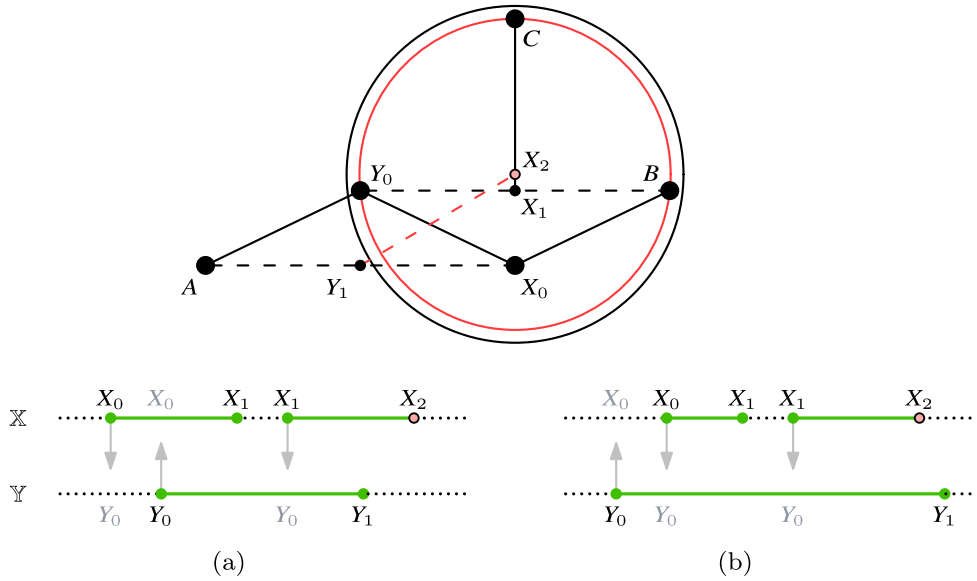

Fig. 4 An example where the (unmodified) Ando algorithm leads to separation in the **a** 1-async model and **b** 2-NESTA model

aggressive implementations of the underlying strategy, and that a more cautious approach would be more successful in avoiding separation. It follows from results developed in Sect. 8 that such modifications are nevertheless doomed to failure in the ASYNC scheduling model.

4.2 Overview of the KKNPS algorithm

The `Look`, `Compute`, and `Move` phases of the KKNPS algorithm take the following form:

Look In the KKNPS algorithm, as in earlier schemes, each robot $\mathbb{Y}$ starts an activity interval by locating, within its local coordinate system, all of the other robots within its visibility range, referred to as *neighbours*. It is not assumed that the visibility radius V is known. Instead, as in [27], in each activity interval, robot $\mathbb{Y}$ computes a (tentative) lower bound $V_{\mathbb{Y}}$ on V, provided by the distance to its current furthest neighbour. Neighbours whose distance is greater than $V_{\mathbb{Y}}/4$ are referred to as *distant* neighbours of $\mathbb{Y}$; others are *close* neighbours. Note that the notions of close and distant are robot-specific and configuration-dependent. In particular (i) $\mathbb{Y}$, by definition, always has at least one distant neighbour; (ii) $\mathbb{X}$ could be a close neighbour of $\mathbb{Y}$ while $\mathbb{Y}$ is a distant neighbour of $\mathbb{X}$; and (iii) a distant neighbour of $\mathbb{Y}$ could, without changing its actual distance, become a close neighbour of $\mathbb{Y}$ (and vice versa).

Compute Robot $\mathbb{Y}$ continues by determining a safe region for motion with respect to each of its neighbours $\mathbb{X}$. Safe regions are designed to ensure that, despite the fact that V is unknown, the connectivity of the initial robot configuration will not be lost; in particular, if robots $\mathbb{X}$ and $\mathbb{Y}$ are mutually visible in their initial configuration (i.e., $|X(0)Y(0)| \leq V$), when both are assumed to be immotile, then, provided both robots continue to confine their movement to the safe regions for motion with respect to the other, their mutual visibility will be maintained thereafter. Even though acquired visibility might be subsequently lost, unlike the schemes of [21, 27], a form of acquired visibility preservation can be shown to hold, which suffices to ensure that there is a point in time after which if robots $\mathbb{X}$ and $\mathbb{Y}$ are even momentarily not mutually visible, then $\mathbb{X}$ and $\mathbb{Y}$ remain separated by distance at least $V/2$. The specification of safe regions, together with their properties that are exploited in subsequent visibility preservation and congregation arguments, is provided in the next subsection; the details of our visibility preservation arguments in models with bounded asynchrony, are developed in Sect. 5.

Move Finally, $\mathbb{Y}$ plans a motion to a target destination contained within the intersection of the safe regions associated with all of its neighbours; the details are set out in Sect. 4.2.2. The target destination in turn is designed to ensure that the convex hull of the robot locations shrinks monotonically, and converges to a point.

Algorithm KKNPS Summarizing, Algorithm KKNPS is sketched in Algorithm 1 from the point of view of a robot $\mathbb{Y}$. It is worth noting that the input is represented by the information acquired during the `Look` phase; the body of the algorithm represents the `Compute` phase; whereas the `Move` phase is simply realized by the movement toward the computed target c.

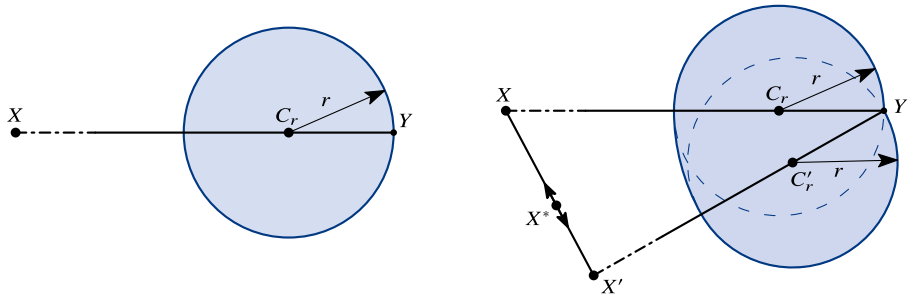

Fig. 5 The region $S_Y^r(X)$ (left) and $S_Y^r(\overline{XX'})$ (right)

Algorithm 1: Algorithm KKNPS performed by any robot $\mathbb{Y}$.

LOOK phase:
1 Visible neighborhood of $\mathbb{Y}$ acquired

COMPUTE phase:
2 Let $\mathbb{X}$ be the neighbor of $\mathbb{Y}$ at maximum distance, and $V_{\mathbb{Y}}$ be such a distance;
3 Let *distant_neighbours* be the set of any neighbor of $\mathbb{Y}$ such that its distance from $\mathbb{Y}$ is at least $V_{\mathbb{Y}}/4$;
4 For each $\mathbb{X} \in distant\_neighbors$, compute $SF(\mathbb{Y}, \mathbb{X})$, i.e. the safe region of $\mathbb{Y}$ with respect to $\mathbb{X}$;
5 $Safe\_region_{\mathbb{Y}} = \bigcap_{\mathbb{X} \in distant\_neighbours} SF(\mathbb{Y}, \mathbb{X})$;
6 Let c be the center of $Safe\_region_{\mathbb{Y}}$;

MOVE phase:
7 *Move* toward c;

4.2.1 Safe regions: specification and properties

Unlike the safe regions used in [21, 27], the safe regions used in the KKNPS algorithm need only be defined with respect to distant neighbours, and depend only on the direction of such neighbours. Let $S_Y^r(X)$ denote the disk, with radius r, centered at the point at distance r from point Y in the direction of point X. Then if $\mathbb{Y}$ is a robot located at Y and $\mathbb{X}$ is a robot located at X, where $|XY| > V_{\mathbb{Y}}/4$, we define the *basic safe region* for motion of $\mathbb{Y}$ with respect to $\mathbb{X}$, to be the region $S_Y^{V_{\mathbb{Y}}/8}(X)$ (cf. Fig. 5 (left)). It is straightforward to confirm that, for any $0 < \alpha \leq 1$, if point P lies within the region $S_Y^r(X)$, then the point P^α at distance $\alpha|PY|$ from Y in the direction of P, lies in $S_Y^{\alpha r}(X)$.

Observation 2 *Safe regions have a very natural generalization to three dimensions: $S_Y^r(X)$ is just the three dimensional ball, with radius r, centered at the point at distance r from point Y in the direction of point X.*

In order to respect the basic safe region of any distant neighbour, of which there must be at least one, a robot can plan a motion of length at most $V_{\mathbb{Y}}/4$. (In fact, as will become clear, when the actual destination point, respecting the safe regions of all neighbours, is specified, a robot will never plan a motion of length greater than $V_{\mathbb{Y}}/8$.) Consequently, if we consider the basic safe region of a robot $\mathbb{Y}$ located at Y, with respect to any close neighbour, to be the disk of radius $V_{\mathbb{Y}}/4$ centred at Y, this provides no additional constraint on the motion of $\mathbb{Y}$. Furthermore, the joint planned motions of close neighbours cannot possibly lead directly to a separation exceeding V.

Note that, if a robot $\mathbb{Y}$ is located in the convex hull of its distant neighbours then the intersection of its safe regions with respect to the distant neighbours contains its current location only, so to respect all of these safe regions it must remain stationary.[15]

In the 1-NESTA and 1-ASYNC models, each robot $\mathbb{Y}$, when activated, is constrained to choose a target location that lies within the basic safe region with respect to each of its neighbours. In the k-NESTA and k-ASYNC models, with $k > 1$, the only change is to simply scale down the basic safe regions by a factor of $1/k$. In this way, robot $\mathbb{Y}$ is constrained to choose a target location that lies within the $1/k$-scaled safe region $S_Y^{V_{\mathbb{Y}}/8k}(X)$ with respect to each distant neighbour robot $\mathbb{X}$, what we refer to as a *fractional move* of *reach* $V_{\mathbb{Y}}/8k$.

In general, when $\mathbb{Y}$ is activated at location Y, any one of its neighbours, say $\mathbb{X}$, might be in transition between two locations, say X and X', so the set of possible target locations for a single fractional move of $\mathbb{Y}$ of reach r is $\bigcup_{X^* \in \overline{XX'}} S_Y^r(X^*)$, which we denote by $S_Y^r(\overline{XX'})$ (cf. Fig. 5 (right)).

The prospect of one robot, $\mathbb{Y}$, at location Y, making up to k successive fractional moves each of reach $V_{\mathbb{Y}}/8k$ while another distant neighbour, $\mathbb{X}$, is in the process of moving from location X to location X', invites the question: how can we characterize the set of points that can be reached by robot $\mathbb{Y}$ in this situation? It is not hard to see that a sequence of $j > 1$ fractional moves each of reach $V_{\mathbb{Y}}/8k$ can take robot $\mathbb{Y}$ beyond $S_Y^{jV_{\mathbb{Y}}/8k}(\overline{XX'})$ but the exact description of this set

[15] It is worth noting that the choice of $V_{\mathbb{Y}}/2$ as the definition of "close" is somewhat arbitrary. Similarly, the size of the safe region associated with distant neighbours has been chosen to have radius $V_{\mathbb{Y}}/8$ mostly for convenience. Anything less than this would certainly work as long as it is at least some positive constant. (Furthermore, choosing a smaller radius would allow us to choose a larger radius for "close").

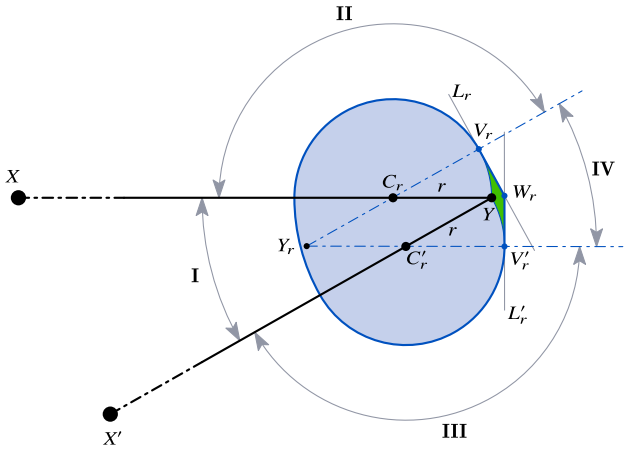

Fig. 6 The region $R_Y^r(\overline{XX'})$, consisting of its core (blue) and bulge green. This is shown to contain all points reachable by robot $\mathbb{Y}$ from location Y in a sequence of fractional moves of total reach r when distant neighbour $\mathbb{X}$ is seen somewhere on the segment $\overline{XX'}$ (colour figure online)

is somewhat complicated, even when $X = X'$. Fortunately, a more simply described superset suffices for our purposes.

We define the region $R_Y^r(\overline{XX'})$ to be the coloured region illustrated in Fig. 6. This figure specifies four sectors, induced by the points X, X' and Y, together with C_r (the point at distance r from Y in the direction of X), C'_r (the point at distance r from Y in the direction of X'), Y_r (the reflection of Y across the segment $\overline{C_r C'_r}$), V_r (the reflection of Y_r across C_r) V'_r (the reflection of Y_r across C'_r), L_r (the line through V_r perpendicular to $\overline{Y_r V_r}$), L'_r (the line through V'_r perpendicular to $\overline{Y_r V'_r}$, and W_r (the point of intersection of L_r and L'_r) that together provide a cover the plane that, like $R_Y^r(\overline{XX'})$, is symmetric with respect to the line through Y and Y_r:

sector I the acute wedge with apex Y and arms through points X and X';
sector II the obtuse wedge with apex C_r and arms through points X and V_r;
sector III the obtuse wedge with apex C'_r and arms through points X' and V'_r; and
sector IV the acute wedge with apex Y_r and arms through points V_r and V'_r.

The region $R_Y^r(\overline{XX'})$ consists of two parts:

the core : the region $S_Y^r(\overline{XX'})$ (shown in blue); and
the bulge : points (shown in green) outside of the core in sector **IV** that are on the same side as Y_r of both L_r and L'_r.

Observation 3 *Note that*

(i) $R_Y^r(\overline{XX'}) = S_Y^r(X)$, *if* X' *is co-linear with* X *and* Y *(in particular, the bulge is empty in this case);*
(ii) *point* Y *has distance* $r(1-\cos\theta)$ *from both lines* L_r *and* L'_r, *where* $\theta = \angle XYX'$; *and*
(iii) $|pq| \leq |pC_r| + r$ *for all points* $p \in \overline{X'Y}$ *and* $q \in R_Y^r(\overline{XX'})$.

As a warm up, we start with the case when robot $\mathbb{X}$ is stationary at point X. In the next lemma we show that region $R_Y^{V/8}(\overline{XX})$, expressed more simply as $R_Y^{V/8}(X)$, contains all the points reachable by robot $\mathbb{Y}$ from position Y by k fractional moves.

Lemma 1 (Circle expansion) *Suppose that* $|YX| \geq V/2$. *Then* $R_Y^{jV/(8k)}(X)$ *contains (strictly, when* $j > 1$*) the set of all points that can be reached by robot* $\mathbb{Y}$ *at location* Y, *making* $1 \leq j \leq k$ *successive fractional moves, each of reach at most* $V/(8k)$, *while another distant neighbour,* $\mathbb{X}$, *remains stationary at location* X.

Proof First, note that, since $|YX| \geq V/2$ by the lemma assumption, and since each move of robot $\mathbb{Y}$ has length at most $V/(4k)$, robot $\mathbb{X}$ never becomes a close neighbour (distance less than $V_{\mathbb{Y}}/4$) of $\mathbb{Y}$. Thus the set of points reachable by j successive fractional moves is exactly the union of the sets $S_{Y^*}^{V/8}(X)$, taken over all Y^* reachable by $j-1$ successive moves.

The proof proceeds by induction on j. The basis ($j = 1$) follows immediately from the fact that $R_Y^r(X)$ contains (in fact, coincides with) $S_Y^r(X)$.

For the inductive step, it suffices to demonstrate the more general fact that $R_Y^{r+s}(X)$ contains the union of $S_{Y^*}^s(X)$ over all $Y^* \in R_Y^r(X)$, provided that $r + s \leq V/8$ and $|XY| \geq 2(r + s)$. This follows from a simple geometric observation captured in Fig. 7. Suppose that point Y^* lies in $R_Y^r(X)$; consider point C_{r+s} at distance s from C_r in the direction to X (see Fig. 7 (left)). Then the centre of the safe region $S_{Y^*}^s(X)$ has distance less than $|Y^*C_r|$ from C_{r+s} (with distance r realized when $|Y^*C_r| = r$ and $X = X^\infty$, where X^∞ denotes the point at infinity in the direction of the ray from Y to X.

It follows that all points in $S_{Y^*}^s(X)$ have distance at most $r+s$ from C_{r+s} (see Fig. 7 (right)), and thus $\bigcup_{Y^* \in S_Y^r(X)} S_{Y^*}^s(X) \subseteq S_Y^{r+s}(X) = R_Y^{r+s}(X)$ if $|XY| \geq 2(r + s)$. As this condition is always satisfied if $|YX| \geq V/2$, the lemma follows. □

More generally, we can show that the region $R_Y^r(\overline{XX'})$ can be used to characterize the set of points reachable from Y by a sequence of moves, when robot $\mathbb{X}$ is in motion.

Lemma 2 (Base region extension) *Suppose that* $|XX'| \leq V/4$ *and* $|YX^*| \geq V/2$, *for all* $X^* \in \overline{XX'}$. *Then*

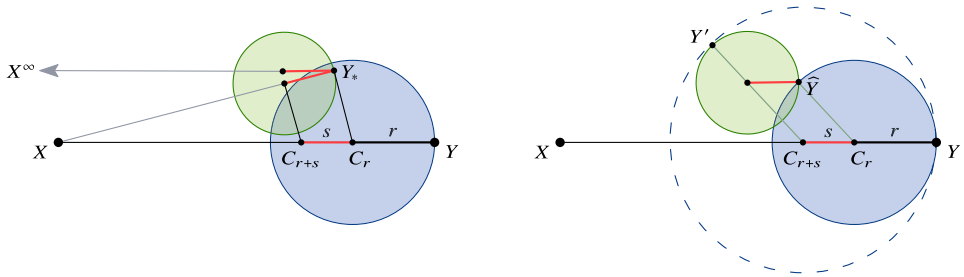

Fig. 7 Circle expansion lemma. Left: Points of $S^s_{Y^*}(X)$ lie within distance $r+s$ from C_{r+s}. Right: $S^{r+s}_Y(X) = \bigcup_{\hat{Y}\in S^r_Y(X)} S^s_{\hat{Y}}(X^\infty)$

$R^{jV/(8k)}_Y(\overline{XX'})$ contains (strictly, when $j \geq 1$) the set of all points that can be reached by robot $\mathbb{Y}$ at location Y, making $1 \leq j \leq k$ successive fractional moves, each of reach at most $V/(8k)$, while another robot, $\mathbb{X}$, is in the process of moving from location X to location X'.

Proof Note that, by assumptions of the lemma, the acute angle $\theta = \angle XYX'$ is at most $\pi/6$. As in Lemma 1, it follows that robot $\mathbb{X}$ never becomes a close neighbour (at distance less than $V_{\mathbb{Y}}/4$) of $\mathbb{Y}$, since each move of robot $\mathbb{Y}$ has length at most $V/(4k)$. Thus the set of points reachable from Y by j successive fractional moves is contained in the union of the sets $S^{V/(8k)}_{Y^*}(\overline{XX'})$, taken over all Y^* reachable by $j-1$ successive moves.

The proof proceeds by induction on j. As in the preceding lemma, the basis follows immediately from the fact that $R^r_Y(\overline{XX'})$ contains $S^r_Y(\overline{XX'})$. Consider a region $Q^{r+s}_Y(\overline{XX'})$ of all the points reachable from $R^r_Y(\overline{XX'})$ by a fractional move of reach s. That is, $Q^{r+s}_Y(\overline{XX'}) = \bigcup_{Y^*\in R^r_Y(\overline{XX'})} S^s_{Y^*}(\overline{XX'})$. For the induction step, it suffices to demonstrate that, assuming $s \leq r$ and $r+s \leq V/8$, the set $R^{r+s}_Y(\overline{XX'})$ contains $Q^{r+s}_Y(\overline{XX'})$.

To reduce the burden of notation, we refer to $S^s_{Y^*}(\overline{XX'})$ as the s-reach of point Y^*. We say that a point $Y^* \in R^r_Y(\overline{XX'})$ is *critical* if the s-reach of Y^* contains a point on the boundary of the convex hull of $Q^{r+s}_Y(\overline{XX'})$. It is straightforward to confirm that no point in the interior of $R^r_Y(\overline{XX'})$ is critical. Thus, to show that $Q^{r+s}_Y(\overline{XX'}) \subseteq R^{r+s}_Y(\overline{XX'})$, it suffices to show that no point outside of $R^{r+s}_Y(\overline{XX'})$ is contained in the s-reach of a point on the boundary of $R^r_Y(\overline{XX'})$.

Let R^+_r be the Minkowski sum of $R^r_Y(\overline{XX'})$ and a disk of radius $2s$ (see Fig. 8). Note that, since the diameter of $S^s_{Y^*}(\overline{XX'})$ is $2s$, no point outside of R^+_r is contained in the s-reach of a point on the boundary of $R^r_Y(\overline{XX'})$ (**Property A**).

Consider the geometric construction of region $R^{r+s}_Y(\overline{XX'})$. In the argument below, we use the same notation for the points and lines introduced when defining the region $R^r_Y(\overline{XX'})$, but replacing the corresponding subscripts by $r+s$ (e.g., C_{r+s}, V_{r+s}, L_{r+s}, etc.). Let $\widehat{L}_{r+s}$ be a line parallel to L_{r+s} passing through the point C_{r+s}. As the distance between L_{r+s} and $\widehat{L}_{r+s}$ is $r+s$, no point beyond L_{r+s} is contained in the s-reach of a point to the left of $\widehat{L}_{r+s}$. (A completely symmetric argument shows that an s-reach of any point to the left of $\widehat{L}'_{r+s}$, the line parallel to L'_{r+s} through C'_{r+s}, does not intersect line L'_{r+s}.)

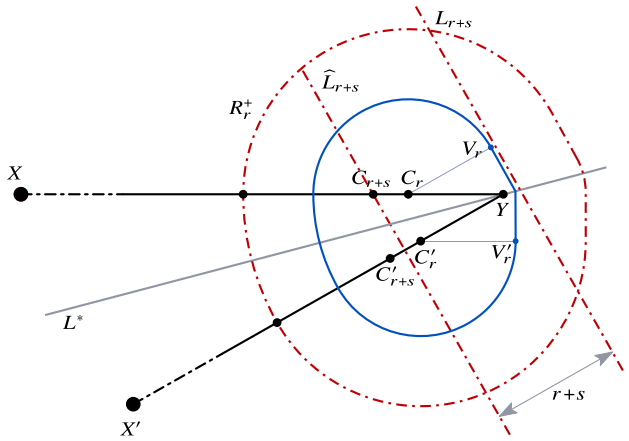

Fig. 8 Proof of Lemma 2. The region $R^r_Y(\overline{XX'})$ (outlined in blue) together with constraints on the s-reach of points on its boundary (colour figure online)

In fact, no point beyond either L_{r+s} or L'_{r+s}, is contained in the s-reach of a point on the boundary of $R^r_Y(\overline{XX'})$ (**Property B**). To show this, it suffices to show that no point beyond L_{r+s} is contained in the s-reach of a point on the boundary of $R^r_Y(\overline{XX'})$, to the right of $\widehat{L}_{r+s}$. The corresponding result for L'_{r+s} follows by symmetry across L^*, the bisector of $\angle XYX'$.

We make essential use of the facts that

(i) for any point Y^* on the boundary of $R^r_Y(\overline{XX'})$, and any point $X^* \in \overline{XX'}$, all points of $S^s_{Y^*}(X^*)$ are at most $s(1-\cos\psi)$ closer than Y^* to L_{r+s}, where ψ denotes the deviation of the ray $\overrightarrow{Y^*X^*}$ from the normal to L_{r+s},
(ii) the maximum deviation from the normal to L_{r+s} is realized by either $\overrightarrow{Y^*X}$ or $\overrightarrow{Y^*X'}$,
(iii) the deviation of $\overrightarrow{Y^*X}$ from the normal to L_{r+s} is less than θ, for all boundary points Y^* above L_{XY}, the line through X and Y, and to the right of $\widehat{L}_{r+s}$, and

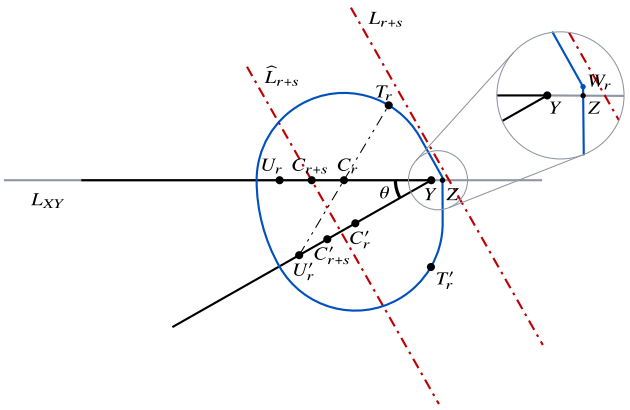

Fig. 9 Proof of Lemma 2. The s-reach of a point Y^* on the boundary of $R^r_Y(\overline{XX'})$ to the right of $\widehat{L}_{r+s}$ does not intersect L_{r+s}

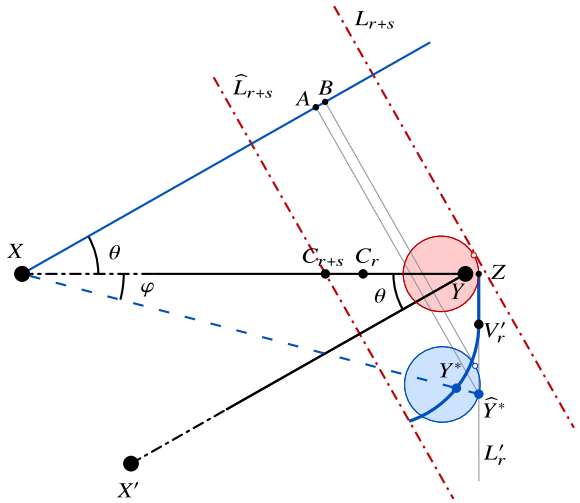

Fig. 10 The third case of Lemma 2

(iv) the deviation from the normal to L_{r+s} realized by $\overrightarrow{Y^*X'}$ is less than the deviation from the normal to L_{r+s} realized by $\overrightarrow{Y^*X}$, for all boundary points Y^* below L_{XY}.

We consider four cases for the location of point Y^* on the boundary of $R^r_Y(\overline{XX'})$ to the right of $\widehat{L}_{r+s}$. First, define U'_r to be the second point at distance r from C_r on the segment $\overline{X'Y}$, and T_r to be the reflection of U'_r across C_r. Similarly, define U_r to be the second point at distance r from C'_r on the segment $\overline{XY}$, and T'_r to be the reflection of U_r across C'_r (see Fig. 9). Note that $|X'Y| > 2r > |U'_rY|$ and $|XY| > 2r > |U_rY|$.

If Y^* lies between $\widehat{L}_{r+s}$ and T_r (inclusive), then, since $|X'Y| > |U'_rY|$, the centre of $S_{Y^*}(X')$ maximizes its distance from X' and minimizes $\angle Y^*X'Y$ (and hence minimizes its distance to L_{r+s}), when $Y^* = T_r$. Similarly, if Y^* lies between $\widehat{L}_{r+s}$ and T'_r, then, since $|XY| > |U_rY|$, the centre of $S_{Y^*}(X)$ maximizes its distance from X and minimizes $\angle Y^*XY$ (and hence minimizes its distance to L_{r+s}), when $Y^* = T'_r$.

Let Z be the intersection point of L_{XY}, the line supporting $\overline{XY}$, and the boundary of $R^r_Y(\overline{XX'})$, and specifically, the segment $\overline{W_rV'_r}$ (see Fig. 9). If Y^* lies between T_r and the point Z (inclusive), then, since $|X'Y| > |U'_rY|$, it follows that $\angle Y^*X'Y$, the deviation of the ray $\overrightarrow{Y^*X'}$ from the normal to L_{r+s}, satisfies $\angle Y^*X'Y < \angle Y^*U'_rY \leq \theta$. Thus, by facts (i) and (iii) above, $S_{Y^*}(\overline{XX'})$ never reaches beyond L_r by more than $s(1-\cos\theta)$, i.e. never as far as L_{r+s}.

If Y^* lies between T'_r and Z, then let φ denote the angle between the line L_{XY} and the line through X and Y^* (see Fig. 10, and note that $\varphi < \theta$). By facts (ii) and (iv) it suffices to show that $S^s_{Y^*}(X)$ does not intersect L_{r+s}. Let $\widehat{Y}^*$ denote the point where the line through X and Y^* intersects L'_r. Since the region $S^s_{\widehat{Y}^*}(X)$ is at least as close to L_{r+s} as $S^s_{Y^*}(X)$, it suffices to show that $S^s_{\widehat{Y}^*}(X)$ lies entirely to the left of L_{r+s}. Note that $|Z\widehat{Y}^*| = |XZ|\tan\varphi$.

Let A denote the normal projection of $\widehat{Y}^*$ onto the line through X normal to L_{r+s}, and let B denote the normal projection of the point of $S^s_{Y^*}(X)$ closest to L_{r+s} onto this same line. Observe that $|AB| = s(1-\cos(\theta+\varphi))$ and $|XA| = |XZ|\cos\theta - |Z\widehat{Y}^*|\sin\theta$. It follows that $\frac{d|XB|}{d\varphi} = s\sin(\theta+\varphi) - |XZ|\frac{\sin\theta}{\cos^2\varphi} < s\sin 2\theta - |XZ|\sin\theta < (2s-|XZ|)\sin\theta < 0$. Hence, $|XB|$ is maximized when $\widehat{Y}^* = Y^* = Z$, where, as we have already seen, $S_{Y^*}(X)$ lies entirely to the left of L_{r+s}. Thus the s-reach of all points on the boundary of $R^r_Y(\overline{XX'})$ lie to the left of L_{r+s}.

Consider the points reachable from $R^r_Y(\overline{XX'})$ in each of the sectors associated with $R^{r+s}_Y(\overline{XX'})$ (see Fig. 11). It is immediate from Property A that the points reachable from $R^r_Y(\overline{XX'})$ in sector **I** form a subset of $R^{r+s}_Y(\overline{XX'})$. Similarly, from Property B it follows that the points reachable from $R^r_Y(\overline{XX'})$ in sector **IV** form a subset of $R^{r+s}_Y(\overline{XX'})$. It remains to show that the same is true for sector **II** (the argument for sector **III** follows by symmetry). To see this, suppose that the region $S^s_{Y^*}(X^*)$, for some $X^* \in \overline{XX'}$, lies inside the circle of radius $2(r+s)$ centred at Y and to the left of L_{r+s}, but contains a point Z in sector **II** at distance greater than $r+s$ from C_{r+s}. Then the centre of $S^s_{Y^*}(X^*)$ must lie in sector **II** at distance greater than r from C_{r+s}. But, as shown in the proof of Lemma 1, this means that Y^* must lie in the obtuse wedge with apex C_r and arms through points X and V_r (i.e. sector **II** associated with $R^r_Y(\overline{XX'})$), at distance greater than r from C_r. Hence, $Y^* \notin R^r_Y(\overline{XX'})$. □

Observation 4 *The characterization of the region of points reachable by a sequence of safe moves given in Lemma 1 extends directly to three dimensions. The corresponding result given in Lemma 2 does not admit such a simple exten-*

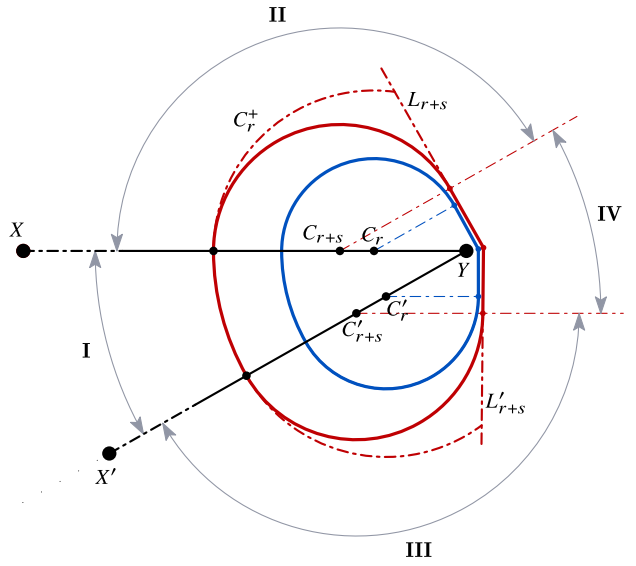

Fig. 11 The region $R_Y^{r+s}(\overline{XX'})$ (outlined in red) contains the set of points reachable by robot $\mathbb{Y}$ in a single fractional move of reach s from points in $R_Y^r(\overline{XX'})$ (blue), when distant neighbour $\mathbb{X}$ is seen somewhere on the segment $\overline{XX'}$ (colour figure online)

sion. Fortunately, this is not needed for the extension of our proof about visibility preservation to three dimensions.

4.2.2 Target destination: specification and properties

To this point, we have not specified the details of the destination function of an activated robot $\mathbb{Y}$, other than that it must be a point in the intersection of the safe regions with respect to all of its distant neighbours (those neighbours whose distance is in the range $(V_{\mathbb{Y}}/2, V_{\mathbb{Y}}]$, where $V_{\mathbb{Y}}$ is the distance to the furthest neighbour of $\mathbb{Y}$). We choose the center of this intersection as the target destination.

In the event that the distant neighbours are not properly contained in any halfspace with $\mathbb{Y}$ on its boundary, the intersection of the corresponding safe regions is just the current location of $\mathbb{Y}$, so the target destination is just this current location (i.e., $\mathbb{Y}$ does not move). Other events consist of two cases:

(i) $\mathbb{Y}$ has only one distant neighbour. In this case the target destination is just the center point of the sole constraining safe region.

(ii) $\mathbb{Y}$ has two or more distant neighbours. In this case the intersection of the safe regions with respect to all of its distant neighbours is determined by the safe regions associated with the two distant neighbours that define the largest sector containing all of the distant neighbours of $\mathbb{Y}$, and the target destination is the center point of the intersection of these two safe regions (see Fig. 12). Note that the target point is the middle point of the segment connecting the centers of the safe regions corresponding to these two extreme distant neighbours.

Note that, in all events, the distance to the target destination of an activated robot is at most $V/8$.

Observation 5 *The "natural" generalization to three dimensions specifies the target to be the closest point (to $\mathbb{Y}$) in the convex hull of the midpoints of the safe regions associated with all distant neighbours of $\mathbb{Y}$.*

5 Visibility preservation by the KKNPS algorithm, under bounded asynchrony

In this section we begin the analysis of the KKNPS algorithm by demonstrating the fact that it guarantees that the connectivity of the initial robot configuration is preserved in both the k-NESTA and k-ASYNC scheduling models. This is established by showing that all pairs of robots that are mutually visible in the initial configuration remain so thereafter. In addition, we demonstrate a limited form of preservation of acquired visibility (mutual visibility of robot pairs that arises in subsequent configurations), critical to our congregation argument.

5.1 Visibility preservation in the k-NESTA model

Though superficially similar, there is an important difference between the SSYNC and 1-NESTA models: in the latter, if an activity interval of robot $\mathbb{Y}$ is nested within an activity interval of robot $\mathbb{X}$, it is possible that the `Look` phase of $\mathbb{Y}$ takes place after the `Move` phase of $\mathbb{X}$ has begun. In this situation, even if we assume that $\mathbb{Y}$ was seen by $\mathbb{X}$ when it last looked, $\mathbb{X}$ could be viewed by $\mathbb{Y}$ anywhere on its chosen trajectory.

For the k-NESTA model, with $k > 1$, demonstrating that mutual visibility of a pair $\mathbb{X}$, $\mathbb{Y}$ of robots is preserved is further complicated by the fact that during repeated activations of $\mathbb{Y}$, nested within one activation of $\mathbb{X}$, $\mathbb{Y}$ may observe $\mathbb{X}$ at many different positions on its determined trajectory. Furthermore, even if $\mathbb{X}$ is stationary at some location X, $\mathbb{Y}$ may view it from up to k different locations (cf. Fig. 13). In fact, (i) as illustrated by the bulge region, k nested activations of $\mathbb{Y}$, each moving within its $1/k$-scaled safe region with respect to $\mathbb{X}$, while preserving mutual visibility, can take $\mathbb{Y}$ outside of its basic (unscaled) safe region, and (ii) k nested activations of $\mathbb{Y}$, each moving within safe regions scaled by something slightly larger than $1/k$ could lead to a break in visibility.

In asynchronous settings, it is certainly possible that a pair of robots under the control of the KKNPS algorithm could become mutually visible, only to have this visibility be subsequently lost. However, a critical component of our congregation argument relies on the fact that, if the separation

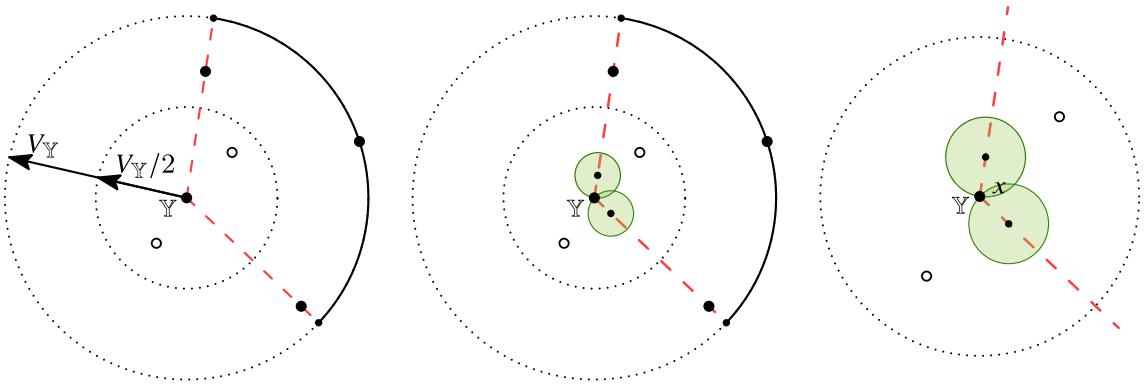

Fig. 12 Visualization of the target destination x for $\mathbb{Y}$, when $\mathbb{Y}$ admits two distant neighbours

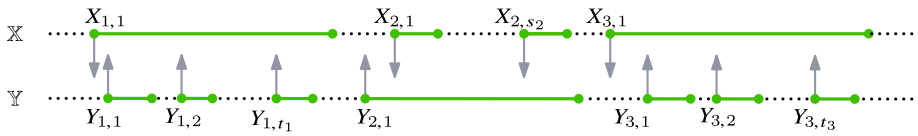

Fig. 13 An example of a k-NESTA timeline for two robots $\mathbb{X}$ and $\mathbb{Y}$. Grey arrows designate the snapshots taken by the robots in the Look phase

of a robot pair becomes sufficiently small, visibility will be preserved indefinitely. We say that robot $\mathbb{X}$ is *strongly visible* from robot $\mathbb{Y}$ at time t if $|X(t)Y(t)| \leq V/2$.

With this definition in hand we can state and prove what we need in terms of visibility preservation for the k-NESTA scheduling model:

Lemma 3 *Provided robots $\mathbb{X}$ and $\mathbb{Y}$ both continue to confine their movement to the $1/k$-scaled safe regions for motion with respect to one another, then if (i) $\mathbb{X}$ is visible from $\mathbb{Y}$ at time $t = 0$ (i.e., $|X(0)Y(0)| \leq V$), or (ii) $\mathbb{X}$ becomes strongly visible from $\mathbb{Y}$ at some subsequent time $t > 0$, then their mutual visibility will be maintained thereafter, under arbitrary k-NESTA scheduling.*

Proof Recall that, in the k-NESTA model, the schedule of activations of any pair of robots $\mathbb{X}$ and $\mathbb{Y}$ decomposes into a sequence of activation events each of which consists of a single activity interval of one of the two robots within which up to k activity intervals of the other are nested (cf. Fig. 13). In fact, there is no loss of generality in restricting attention to *full* activation events in which the activity interval of one robot is completely partitioned by the activity intervals of the other. The only exception is the initial activation event, which may begin with an interval during which one robot is active before the other becomes active (but mutual visibility is sustained). Thus it will suffice to prove that (i) if $\mathbb{X}$ and $\mathbb{Y}$ are mutually visible at the start of any activation event, then they remain mutually visible throughout the event, and (ii) if one of the robots becomes strongly visible from the other during the course of the activation event, then they will remain mutually visible thereafter. In fact, since every fractional move has length at most $V/(8k)$, the total length of moves of $\mathbb{X}$ and $\mathbb{Y}$ in any activation event is at most $V/4$. Thus one robot can become strongly visible from the other during the course of an activation event only if the robots are mutually visible at both the start and end of that event. Hence, it suffices to prove property (i) alone, knowing that at no intermediate point one of the robots becomes strongly visible from the other.

Suppose that an activation event begins with the pair of robots $\mathbb{X}$ and $\mathbb{Y}$ at locations X_0 and Y_0 respectively, with separation at most V. Suppose further that this activation event consists of a single activity interval of robot $\mathbb{X}$ and some $j \leq k$ activity intervals of robot $\mathbb{Y}$ all of which are nested within the activity interval of $\mathbb{X}$ (in particular, at both the start and end of the activity interval of $\mathbb{X}$, both $\mathbb{X}$ and $\mathbb{Y}$ are immotile). Finally, suppose that (i) the activity interval of robot $\mathbb{X}$ determines a target destination X_1 that is confined to the $1/k$-scaled safe region $S_{X_0}^{V_{\mathbb{X}}/(8k)}(Y_0)$, and (ii) the ith activity intervals of robot $\mathbb{Y}$ determines a target destination Y_i that is confined to the $1/k$-scaled safe region $S_{Y_{i-1}}^{V_{\mathbb{Y}}/(8k)}(X^*)$ for some $X^* \in \overline{X_0X_1}$.

Now suppose, leading to a contradiction, that at the end of the jth activity interval of robot $\mathbb{X}$, robots $\mathbb{X}$ and $\mathbb{Y}$ are no longer mutually visible. In this case, we can assume that we are dealing with a counterexample with minimum j. Since, by assumption, the robots are never closer than $V/2$, it follows from Lemma 2 that the set of points reachable by robot $\mathbb{Y}$ from location Y_0 is contained in $R_{Y_0}^{jV/(8k)}(\overline{X_0X_1})$

But, by Observation 3, it follows that among points in $R_{Y_0}^{jV/(8k)}(\overline{X_0X_1})$ the distance to X_1 is at most $|X_1C_0| + jV/(8k)$, where C_0 denotes the centre of $S_{Y_0}^{jV/(8k)}(X_0)$. Since $|X_1C_0| \leq |X_0C_0| < V - jV/(8k)$, it follows that

$|X_1P| < V$, for all points $P \in R_{Y_0}^{jV/(8k)}(\overline{X_0X_1})$, contradicting our assumption that robots $\mathbb{X}$ and $\mathbb{Y}$ are no longer mutually visible at the end of the current activity interval of robot $\mathbb{X}$. □

5.2 Visibility preservation in the *k*-ASYNC model

Turning now to the more inclusive k-ASYNC scheduling model, we establish the following generalization of Lemma 3, exploiting the structure common to the k-NESTA and k-ASYNC models.

Lemma 4 *Provided robots $\mathbb{X}$ and $\mathbb{Y}$ both continue to confine their movement to the $1/k$-scaled safe regions for motion with respect to one another, then if (i) $\mathbb{X}$ is visible from $\mathbb{Y}$ at time $t = 0$ (i.e., $|X(0)Y(0)| \leq V$), or (ii) $\mathbb{X}$ becomes strongly visible from $\mathbb{Y}$ at some subsequent time $t > 0$, then their mutual visibility will be maintained thereafter, under arbitrary k-ASYNC scheduling.*

Proof As in our proof of Lemma 3, we start by showing that initial visibilites are preserved. To this end suppose that robots $\mathbb{X}$ and $\mathbb{Y}$ have separation at most V in their initial configuration (when both are immotile). We need to show that, from this point onward, provided robots $\mathbb{X}$ and $\mathbb{Y}$ both continue to confine their movement to the $1/k$-scaled safe regions for motion with respect to the other, they will remain mutually visible thereafter.

Note that the argument here seems to be intrinsically more difficult than that used in the k-NESTA model. In particular, it does not suffice, as it did in the k-NESTA model, to focus attention on just one activity interval for one of the two robots whose mutual visibility is being considered. Even in the 1-ASYNC model, the validity of each step depends on the full history of transitions since the last time both robots were simultaneously immotile.

We say that a pair of robots $\mathbb{X}$ and $\mathbb{Y}$ is *engaged* at time t if at least one of them is active. Suppose that $\mathbb{X}$ and $\mathbb{Y}$ become engaged at time t_0, with $|X(t_0)Y(t_0)| \leq V$. We proceed to show that while $\mathbb{X}$ and $\mathbb{Y}$ remain engaged their separation remains at most V. Suppose, leading ultimately to a contradiction, that a sequence of activations of robots $\mathbb{X}$ and $\mathbb{Y}$ exists that results in separation exceeding V. We consider a minimal such separating sequence, one that minimizes the total number of activations. Without loss of generality we can assume:

1. At time t_0, $\mathbb{X}$ is starting an activity interval and $\mathbb{Y}$ is immotile, i.e., either idle or starting an activity interval,
2. We refer to the sequence of up to k activations of one robot that take place during a single activity interval of the other as an *activation cluster*. Throughout their engagement, activation clusters of $\mathbb{X}$ alternate with those of $\mathbb{Y}$. Accordingly, we denote by $X_{j,1}, \ldots, X_{j,s_j}$, where $s_j \leq k$, the positions of robot $\mathbb{X}$ at the start of the activity intervals in its jth activation cluster, which occurs as robot $\mathbb{Y}$ is executing the last activity interval of its $(j-1)$st activation cluster. Similarly, we denote by $Y_{j,1}, \ldots, Y_{j,t_j}$, where $t_j \leq k$, the positions of robot $\mathbb{Y}$ at the start of the activity intervals in its jth activation cluster, which occurs as robot $\mathbb{X}$ is executing the last activity interval of its jth activation cluster (see Fig. 14).
3. We refer to the locations at the start of the first activity interval in each activation clusters as *activation checkpoints*. For example, $X_{2,1}$ is a checkpoint of $\mathbb{X}$ corresponding to the start of the robot's second activation cluster. There is no loss of generality in assuming that the first checkpoint at which a separation occurs is when robot $\mathbb{X}$ is at location $X_{i,1}$, and that at this time $\mathbb{Y}$ has reached $Y_{i,1}$.
4. Each move of $\mathbb{X}$ and $\mathbb{Y}$, the timing and extent of which is determined by an adversarial scheduler, has a target destination that respects the safe regions of $\mathbb{X}$ and $\mathbb{Y}$ alone (and hence may form a strict superset of the realizable destinations when other robots are taken into consideration). Specifically, the following *safety constraints* hold, for all $j < i$,
   (i) for $1 \leq t < t_j$, the point $Y_{j,t+1}$ lies in the $1/k$-fraction safe region associated with $\mathbb{Y}$ at location $Y_{j,t}$, viewing $\mathbb{X}$ at some location between X_{j,s_j} and $X_{j+1,1}$, and
   (ii) the point $Y_{j+1,1}$ lies in the $1/k$-fraction safe region associated with $\mathbb{Y}$ at location Y_{j,t_j}, viewing $\mathbb{X}$ at some location between X_{j,s_j} and $X_{j+1,1}$.

   Similarly,

   (iii) for $1 \leq s < s_j$, the point $X_{j,s+1}$ lies in the $1/k$-fraction safe region associated with $\mathbb{X}$ at location $X_{j,s}$, viewing $\mathbb{Y}$ at some location between $Y_{j-1,s_{j-1}}$ and $Y_{j,1}$, and
   (iv) the point $X_{j+1,1}$ lies in the $1/k$-fraction safe region associated with $\mathbb{X}$ at location X_{j,s_j}, viewing $\mathbb{Y}$ at some location between $Y_{j-1,s_{j-1}}$ and $Y_{j,1}$.
5. Thus, by Lemma 2, for all $1 \leq j < i$, the checkpoint $Y_{j+1,1}$ lies in the region $R_{Y_{j,1}}^{V/8}(\overline{X_{j,1}X_{j+1,1}})$, and the checkpoint $X_{j+1,1}$ lies in the region $R_{X_{j,1}}^{V/8}(\overline{Y_{j-1,1}Y_{j,1}})$.

We analyse the sequence of checkpoints in reverse chronological order:

$$Y_{i,1},\ X_{i,1},\ Y_{i-1,1},\ X_{i-1,1}, \ldots, X_{2,1},\ Y_{1,1},\ X_{1,1},\ Y_{0,1},\ X_{0,1},\ Y_{-1,1}$$

associated with the minimal separating sequence for $\mathbb{X}$ and $\mathbb{Y}$. This essentially corresponds to a walk from the terminal checkpoint back to the initial checkpoint, ultimately showing

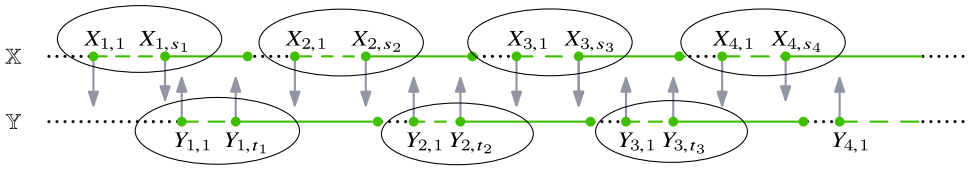

Fig. 14 Without loss of generality, activity intervals for robots $\mathbb{X}$ and $\mathbb{Y}$ in any minimal separating sequence interleave, up to disconnection, with at most k activations of one between activations of the other. Activation clusters are circled

that no such sequence exists that satisfies all of the specified constraints, thereby demonstrating our contradiction.

For $t \geq 0$, let e_t denote the tth edge on the chain formed by successive checkpoints; i.e., $e_t = \overline{Y_{i-t/2,1}X_{i-t/2,1}}$, for even t, and $e_t = \overline{X_{i-(t-1)/2,1}Y_{i-1-(t-1)/2,1}}$, for odd t. Let θ_t denote the angle, in the range $[0, \pi]$, between e_t and e_{t+1}. Note that $\theta_{2i} = 0$.

It is not hard to see that, in order to satisfy local safety constraints, there is a kind of tradeoff between $|e_t|$ and θ_t: specifically, if $|e_t|$ is increased then the safety constraint at the common endpoint of e_t and e_{t+1} can be satisfied with a smaller angle θ_t. We make this tradeoff explicit in the following assertion, which we prove in Lemma 5 below:

$$|e_t| > V\cos\theta_t \text{ and } \cos\theta_t \geq \sqrt{35/36} \approx 0.986, \text{ for all } t \geq 0.$$

Given this assertion, we reach the contradiction that the edge e_{2i} has length greater than V, since $\theta_{2i} = 0$, completing the proof that the initial visibility between $\mathbb{X}$ and $\mathbb{Y}$ must be preserved.

Furthermore, since (i) any checkpoint configuration of $\mathbb{X}$ and $\mathbb{Y}$ leading to a checkpoint separation of at least V must be preceded by checkpoint configurations of $\mathbb{X}$ and $\mathbb{Y}$ with separation at least $.986V$, and (ii) between checkpoint configurations the separation of $\mathbb{X}$ and $\mathbb{Y}$ cannot change by more than $V/4$, it follows immediately that any acquired strong visibility also results in mutual visibility being preserved thereafter.

Thus, to complete the proof of Lemma 4 it only remains to prove Lemma 5 □

Lemma 5 *$|e_t| > V\cos\theta_t$ and $\cos\theta_t \geq \sqrt{35/36} \approx 0.986$, for all $t \geq 0$.*

Proof The proof proceeds by induction on t. Since $|e_0|$, the length of edge $\overline{X_{i,1}Y_{i,1}}$, is at least V, it follows immediately that $|e_0| \geq V\cos\theta_0$. To complete the basis of our induction, observe that in order to satisfy the safety constraint at $Y_{i-1,1}$ it must be the case that $Y_{i-1,1}$ is no further than $V/8$ from $Y_{i,1}$, and hence no further than $V/8$ from e_0. It follows that $|e_0|\sin\theta_0 \leq V/8$, and thus $\sin\theta_0 \leq 1/8$, which holds exactly when $\cos\theta_0 \geq \sqrt{63/64} > \sqrt{35/36}$.

Suppose now that $|e_t| > V\cos\theta_t$ and $\cos\theta_t \geq \sqrt{35/36}$. We will show that $|e_{t+1}| > V\cos\theta_{t+1}$ and $\cos\theta_{t+1} \geq \sqrt{35/36}$. For this induction step, consider four successive checkpoints (renamed P, Q, R, and S, for simplicity and to reflect the commonality of the argument for even and odd t), the associated edges $\overline{PQ}$, $\overline{QR}$ and $\overline{RS}$, and their intervening angles (renamed φ and β) (see Fig. 15). We show that assuming $|SR| > V\cos\beta$ and $\cos\beta \geq \sqrt{35/36}$, then $|RQ| > V\cos\varphi$ and $\cos\varphi \geq \sqrt{35/36}$.

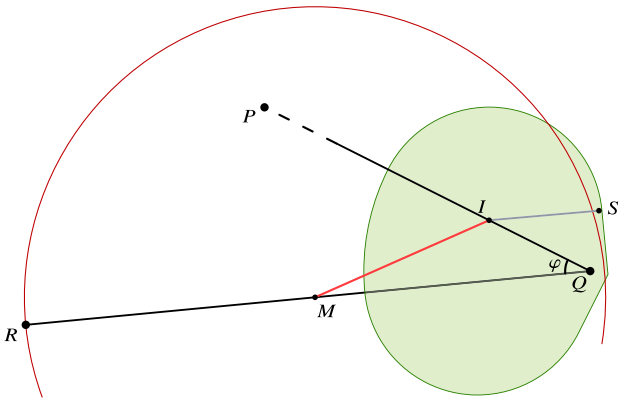

Fig. 15 The point M is at distance $V/2$ from R, and is the center of the circle $c_{V/2}$(in red). The safe region $R_Q^{V/8}(\overline{PR})$) is shown in green. The point S must lie outside $c_{V/2}$ for $|SR| > V\cos\beta$ to hold (colour figure online)

First observe that motion from P to R is the result of a single move step, and the motion from Q to S is the result of single activation cluster, consisting of at most k move steps. As we have seen, this means that $|PR|$ and $|QS|$ are both at most $V/8$. It follows then that since $|RS| \geq \sqrt{35/36}V > 3V/4$, $|P^*Q^*| \geq |RS| - V/4 > V/2$ for all points P^* on $\overline{PR}$ and all points Q^* on the path traced by the robot at location Q in its motion from location Q to location S. Thus the robot at location Q sees the other robot as a distant neighbour throughout its current activation cluster.

Consider a circle $c_{V/2}$ of radius $V/2$ centered at a point M lying on the segment $\overline{RQ}$ at distance $V/2$ from R (shown in red in Fig. 15). Point S must be outside of $c_{V/2}$, otherwise $|SR| \leq V\cos\beta$ (due to the angle $\angle RSQ' \geq \pi/2$ where Q' is an intersection of the extension of $\overline{RQ}$ with $c_{V/2}$); hence $|MS| > V/2$. Let I be the point on $\overline{QP}$ at distance $V/8$ from Q. By Observation 3, $|MS| \leq |MI| + V/8$, and hence $|MI| > 3V/8$.

Consider the triangle $\triangle QIM$. From the cosine theorem it follows that

$$|MI|^2 = |MQ|^2 + |IQ|^2 - 2|MQ||IQ|\cos\varphi.$$

Then, after dividing by V^2 and replacing $|MQ|/V$ by x, the equation above becomes

$$x^2 - 2x\frac{|IQ|}{V}\cos\varphi + \left(\frac{|IQ|}{V}\right)^2 - \left(\frac{|MI|}{V}\right)^2 = 0.$$

Since $x > 0$, it follows that $x = \frac{|IQ|}{V}\cos\varphi + \sqrt{\left(\frac{|IQ|}{V}\right)^2\cos^2\varphi - \left(\frac{|IQ|}{V}\right)^2 + \left(\frac{|MI|}{V}\right)^2}$. But since $|IQ| = V/8$, $|MI| > 3V/8$, and $\cos\varphi \le 1$, we get

$$x > \frac{\cos\varphi}{8} + \sqrt{\frac{\cos^2\varphi}{64} - \frac{1}{64} + \frac{9}{64}}$$
$$\ge \frac{\cos\varphi}{8} + \sqrt{\frac{\cos^2\varphi}{64} + \frac{\cos^2\varphi}{8}} = \frac{\cos\varphi}{2}.$$

Thus, $\frac{|RQ|}{V} = \frac{1}{2} + \frac{|MQ|}{V} > \frac{1}{2} + \frac{\cos\varphi}{2} \ge \cos\varphi$.

Now, as $|PR| \le V/8$ and the length of the perpendicular dropped from R onto the line supporting $\overline{PQ}$ is at most $|PR|$, we obtain that $\sin\varphi \le \frac{V}{8|QR|} < 1/6$ or $\cos\varphi > \sqrt{35/36}$. □

Observation 6 *Lemma 4 extends to three dimensions. To prove this it is not necessary to formulate a three dimensional analogue of the region $R^r_{\mathbb{Y}}(\overline{XX'})$. Instead, more general visibility preservation can be proved by reduction to the two-dimensional result: we can show that if a separating sequence for a pair of robots exists in three dimensions then one must also exist in two dimensions, contrary to Lemma 4. The idea, expanded in Sect. 7.4.2, is that any three dimensional separating sequence can be "flattened" while preserving all of the associated safety constraints as well as all distances between activation checkpoints.*

6 Incremental congregation in the *k*-NESTA and *k*-ASYNC models

As we have demonstrated, the KKNPS algorithm has the property that it preserves all initial visibilities as well as all visibilities that at some point fall below the $V/2$ threshold. We say that a pair of robots are *strong neighbours* at some point in time if one of these conditions holds. It follows that the graph of the strong neighbour relation is both connected, because it includes all initially visible pairs, and monotonic. In particular, at any moment in time there is a path in this graph joining any pair of robots. Furthermore there is a point in time after which the graph no longer changes: thereafter for any pair of robots, (i) their separation is continuously less than or equal to V (i.e., they are mutually visible), or (ii) their separation is continuously greater than $V/2$. This, combined with the fact that our strategy is hull-diminishing, is used to establish the following theorem, the first of our primary results.

Theorem 1 *The* KKNPS *algorithm solves the* COHESIVE CONVERGENCE *task for an arbitrary number of robots moving in the plane under the k-*ASYNC *scheduling model, starting from any connected configuration.*

We begin with an informal overview of the argument.

- As with all hull-diminishing strategies, progress towards convergence to a point can be measured in terms of the shrinkage of the convex hull of the full set of robot locations. Let CH_t denote the convex hull of the robot locations, including their planned but as yet unrealized trajectories, at time t. Then incremental progress towards convergence is captured by the following observation (made previously in [27]):

  $CH_{t^+} \subseteq CH_t$ for all $t^+ > t$. This follows from the facts that (i) when a motion is planned it is towards a point inside the current convex hull, and (ii) when the motion completes, and the planned trajectory (or some prefix) has been realized, replacing the trajectory by its endpoint can only shrink the convex hull.

- Since convex hulls form a nested sequence, they either converge to a single point (which is what we are trying to establish), or not. Suppose that the convex hull sequence does not converge in this way for some initial configuration of n robots. Then both the length of the convex hull perimeter and the hull radius (the radius of the smallest ball enclosing the convex hull) must form monotonically decreasing sequences that converge to some non-zero values, Λ and ρ respectively. Thus, for any $\varepsilon > 0$ there is a time t_ε after which the hull perimeter lies in the range $[\Lambda, \Lambda+\varepsilon)$ and the hull radius lies in the range $[\rho, \rho+\varepsilon)$.
- Let t be any time after which the hull radius lies in the range $[\rho, 3\rho/2)$ and consider the convex hull $H = CH_t$, with perimeter length Λ_t, and its smallest bounding circle Ξ, with center O_H and radius r_H, at that time (see Fig. 16). We will argue that there is a $\lambda > 0$, dependent on ρ, such that at some time $t' > t$ the perimeter length of $CH_{t'}$ becomes less than $\Lambda_t - \lambda$. This leads to a contradiction, if we choose $\varepsilon < \lambda$ and $t \ge t_\varepsilon$.

We know that Ξ is determined by three critical points (robot positions) A_H, B_H and C_H (two of which might coincide), all of which are at distance exactly r_H from O_H, and in pairs bound sectors of Ξ with span no more than π. We will show that there is a $\delta > 0$, dependent on r_H, such that for at

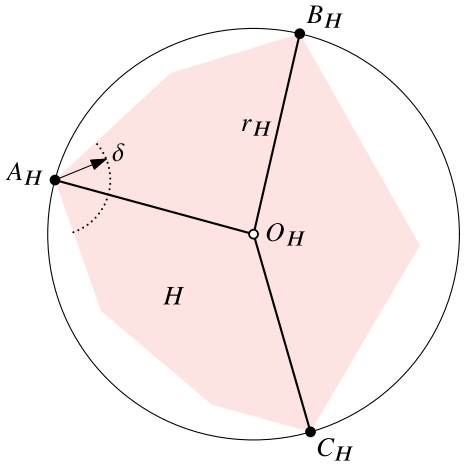

Fig. 16 Smallest bounding circle Ξ

least one of these critical points, say A_H, $\Gamma_\delta(A_H)$, the set of robots at distance at most δ from A_H, is eventually empty.

Clearly, every robot $\mathbb{Z}$ has $V_\mathbb{Z} \leq 2r_H$. The more critical observation is captured in a lemma that states that the motion of any robot $\mathbb{Z}$ with $V_\mathbb{Z} > 0$ cannot take $\mathbb{Z}$ to a point too close to A_H. From this, it follows immediately that if $\mathbb{Z}$ continues to have at least one distant neighbour that is bounded away from A_H then $\mathbb{Z}$ must become and stay outside $\Gamma_\delta(A_H)$.

Lemma 6 *Suppose $V_\mathbb{Z} \geq \nu \cdot r_H$, where $\nu > 0$. Then any ξ-rigid motion of $\mathbb{Z}$ takes it to a point at distance at least $\frac{\nu^2\xi^2}{512} r_H$ from A_H.*

Proof Refer to Fig. 17 (left). Suppose that $V_\mathbb{Z} \geq \nu \cdot r_H$. Let $d = \frac{\nu^2\xi^2}{512} r_H$, and consider a circle $\Gamma_d(A_H)$ of radius d centered at A_H. Let y_d be the length of the chord $P_d P'_d$ (shown in blue), tangent to $\Gamma_d(A_H)$ and perpendicular to $A_H O_H$, and let R_d denote the portion of Ξ, containing point A_H, between the arc $P_d P'_d$ and the chord $P_d P'_d$. Note that $y_d = 2\sqrt{r_H^2 - (r_H - d)^2} < 2\sqrt{2r_H d}$. Let T denote the target destination of robot $\mathbb{Z}$, and T' the point at distance $\xi|XT|$ from X in the direction of T. Suppose, leading to a contradiction of our assumption that $V_\mathbb{Z} \geq \nu \cdot r_H$, that some point on the segment $\overline{T'T}$ has distance less than d from A_H. We consider two cases: (i) the point Z lies outside of R_d, and (ii) the point Z lies inside of R_d.

In the first case, we can assume that $T \in R_d$; otherwise the entire segment $\overline{ZT}$ lies outside R_d and hence the entire motion of $\mathbb{Z}$ remains at distance greater than d from A_H. Since T is the midpoint of safe region centers of two (or one) distant neighbours of $\mathbb{Z}$, the safe region center C_W associated with at least one extreme distant neighbour $\mathbb{W}$ of $\mathbb{Z}$, as well as the location Z of $\mathbb{Z}$, must also lie in R_d. Hence, $|ZW| = |ZC_W| + |C_W W| < V_\mathbb{Z}/8 + y_d$, and since $\mathbb{W}$ is a distant neighbour of $\mathbb{Z}$, $V_\mathbb{Z} < 2|ZW| < 2(V_\mathbb{Z}/8 + y_d)$. Thus $V_\mathbb{Z} < 8/3 y_d$.

In the second case, the centre C_W of the safe region of at least one of the distant neighbours $\mathbb{W}$ of $\mathbb{Z}$ must lie inside the region $R_{d/\xi}$ bounded by the chord $P_{d/\xi} P'_{d/\xi}$ Otherwise the entire segment $\overline{T'T}$ lies outside of R_d and so the ξ-rigid motion of $\mathbb{Z}$ leaves it at distance at least d from A_H. Since $|P_{d/\xi} P'_{d/\xi}| \leq y_d/\xi$, it follows that $|ZC_W| < y_d/\xi$ and hence $V_\mathbb{Z} < 8y_d/\xi$.

In both cases, we have $V_\mathbb{Z} < 8y_d/\xi$ (since $\xi < 1$) and so $V_\mathbb{Z} < 16\sqrt{2r_H d}/\xi < \nu r_H$, contradicting our assumption. □

It follows directly from Lemma 6 that permanent separation from A_H is contagious:

Lemma 7 *If robot $\mathbb{Z}$ has a neighbour $\mathbb{X}$ whose distance from A_H remains at least $\nu \cdot r_H$, assuming ξ-rigid motions, then $\mathbb{Z}$ must itself become and remain at distance at least $\frac{\nu^2\xi^2}{4608} r_H$ from A_H.*

Proof By Lemma 6, while $V_\mathbb{Z} \geq (\nu/3) r_H$, any ξ-rigid motion of $\mathbb{Z}$ must leave it at distance at least $\frac{\nu^2\xi^2}{4608} r_H$ from A_H. On the other hand, if $V_\mathbb{Z} < (\nu/3) r_H$ then, since $\mathbb{X}$ is a neighbour of $\mathbb{Z}$, $\mathbb{Z}$ must have distance at least $(2\nu/3) r_H$ from A_H. But any movement of $\mathbb{Z}$ must leave it no further than $(\nu/3) r_H$ from its current position, and hence at least $(\nu/3) r_H > \frac{\nu^2\xi^2}{4608} r_H$ from A_H. □

Observation 7 *Lemmas 6 and 7 above hold without change in three dimensions.*

Suppose now that we have chosen a time t after which the strong neighbour graph no longer changes and the hull radius lies in the range $[\rho, 3\rho/2)$. Let $\delta = \left(\frac{\xi^2}{64 \cdot 512 n^2}\right)^2 \frac{\xi^2}{4608} r_H$ and consider robots in $\Gamma_\delta(A_H)$. Consider some time $t' > t$ after which all robots that will eventually lie and remain outside of $\Gamma_\delta(A_H)$ have done so. We argue that $\Gamma_\delta(A_H)$ is in fact empty at time t'. Supposing otherwise, let $\mathbb{Z}$ be one of the robots in $\Gamma_\delta(A_H)$ at time t'. It will suffice to argue that, assuming motion is ξ-rigid, $\mathbb{Z}$ must move and remain outside $\Gamma_\delta(A_H)$.

There are two cases to consider depending on the composition of the set of neighbours of $\mathbb{Z}$.

(i) All other robots are neighbours of $\mathbb{Z}$. In this case, either (i) at some subsequent point in time, either one of $\Gamma_\delta(A_H)$, $\Gamma_\delta(B_H)$ or $\Gamma_\delta(C_H)$ contains no robots (in which case nothing remains to be proved), or (ii) $\mathbb{Z}$ must continue to have a neighbour at distance at least $r_H - 2\delta$, that is $V_\mathbb{Z} \geq r_H - 2\delta > r_H/2$, since $\delta < r_H/4$. Hence, by Lemma 6, $\mathbb{Z}$ must move to, and remain at, a position at least $\frac{\xi^2}{2048} r_H$ from A_H.

(ii) There is at least one other robot that is not a neighbour of $\mathbb{Z}$. In this case, since the strong neighbour graph is connected, there must be a strong neighbour $\mathbb{X}$ of $\mathbb{Z}$ that has a strong neighbour $\mathbb{Y}$ that is not a strong neighbour of

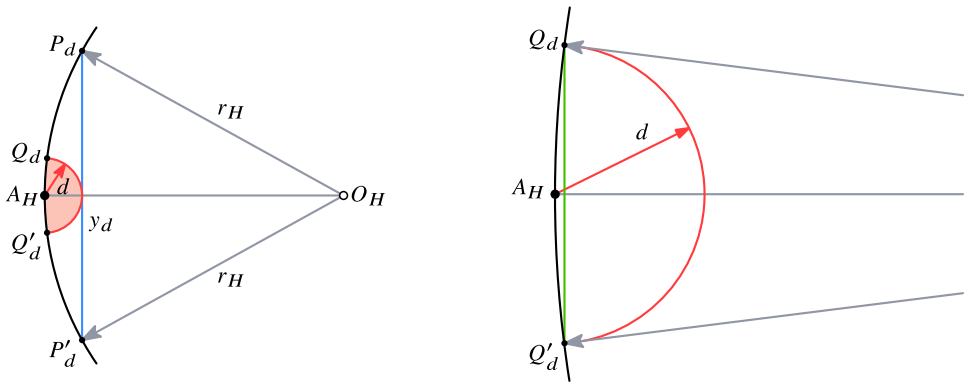

Fig. 17 Left: the radius of the red circle is d, and the length of the chord $\overline{P_d P'_d}$ (in blue) is y_d. Right: The length of the chord $\overline{Q_d Q'_d}$ (in green) is $2z_d$, and the distance from A_H to $\overline{Q_d Q'_d}$ is x_d (colour figure online)

$\mathbb{Z}$. Since $\mathbb{Y}$ and $\mathbb{Z}$ are not strong neighbours their separation must continue to exceed $V/4$, and hence the distance from at least one of them to $\mathbb{X}$ must continue to be at least $V/8$ which, by the connectivity of the strong neighbour graph, is at least $r_H/(8n)$. Thus, $V_{\mathbb{X}} > r_H/(8n)$ and so, by Lemma 6, $\mathbb{X}$ itself must move to, and remain at, a position at least $\frac{\xi^2}{64\cdot 512n^2} r_H$ from A_H. It follows, by Lemma 7, that $\mathbb{Z}$ too must move to, and remain at, a position at least $\left(\frac{\xi^2}{64\cdot 512n^2}\right)^2 \frac{\xi^2}{4608} r_H$ from A_H.

Hence, since $\delta = \left(\frac{\xi^2}{64\cdot 512n^2}\right)^2 \frac{\xi^2}{4608} r_H$ $\mathbb{Z}$ must eventually, and permanently, vacate $\Gamma_\delta(A_H)$. This contradicts our assumption that at time t' all robots that will permanently vacate $\Gamma_\delta(A_H)$ have done so. Thus, we can assume that at time t', and thereafter, $\Gamma_\delta(A_H)$ contains no robots.

Since $CH_{t'}$, the convex hull of the robot positions at time t', has no vertex closer than δ to A_H, it follows that the boundary length of $CH_{t'}$ is less than Λ_t, the boundary length of CH_t, by an amount that is independent of ε.

Lemma 8 *If $\Gamma_d(A_H)$ is empty at time t', then the boundary length of $CH_{t'}$ is at most $\Lambda_t - \frac{d^3}{(2r_H)^2}$.*

Proof Since A_H lies on the boundary of CH_t, and all robot positions at time t' lie outside of $\Gamma_d(A_H)$, it follows that the perimeter of $CH_{t'}$ is at least $2(d - z_d)$ shorter than the perimeter of CH_t, where $2z_d$ is the length of the chord $\overline{Q_d Q'_d}$ (see Fig. 17 (right)). Let x_d be the distance from A_H to $\overline{Q_d Q'_d}$. It is straightforward to confirm that $d - z_d = x_d^2/(d + z_d) > x_d^2/(2d)$ and (by similar triangles) $x_d = d^2/(2r_H)$, and hence $2(d - z_d) > d^3/(2r_H)^2$. □

As previously noted, this leads to a contradiction (of our assumption of non-convergence) if we choose $\varepsilon < \frac{d^3}{(2r_H)^2}$, where $d = \left(\frac{\xi^2}{64\cdot 512n^2}\right)^2 \frac{\xi^2}{4608} r_H$, and $t > t_\varepsilon$. Therefore the sequence of nested convex hulls must converge to a point, completing the proof of Theorem 1

Observation 8 *A result analogous to Lemma 8 holds for robot configurations in three dimensional space as well. In three-dimensions either (i) the surface area of $CH_{t'}$ decreases by an amount that is independent of ε, or (ii) the radius of the smallest enclosing ball of $CH_{t'}$ decreases by an amount that is independent of ε (see Sect. 7.4.2 for details). In either case, if we choose ε sufficiently small, it follows that the sequence of nested convex hulls must converge to a point, thereby showing that Theorem 1 holds for robots moving in three-dimensional space as well.*

7 Extensions/generalizations

7.1 The parameter *k* known only approximately

In certain settings, the parameter k (the bound on potential asynchrony) may not be known precisely. One such scenario is where the duration of all activity intervals is bounded from above and (away from zero) below. Nevertheless, the KKNPS algorithm works correctly with any conservative upper bound on k at the cost of slowing down convergence in a proportional way.

7.2 Error tolerance

We have already observed that, in the absence of any global information on the initial configuration, even an arbitrarily small amount of absolute error in distance or angle measurements, or in the actual destination relative to the trajectory to the intended destination, cannot be tolerated by algorithms that solve Point Convergence, even in fully synchronous scheduling environments. Nevertheless, the KKNPS algorithm can be modified to tolerate a modest amount of relative error.

- The only concern with error in distance measurements is that it might lead $V_{\mathbb{Z}}$, the perceived distance to the furthest neighbour of $\mathbb{Z}$, to become an overestimate of V. If the error is bounded by some small fraction δ of the true distance, then this can be avoided by simply scaling the perceived distance by a factor of $1/(1+\delta)$.
- The KKNPS algorithm can also be modified to tolerate a modest amount of a kind of relative error in angle measurements (that preserves sidedness with respect to lines through neighbouring points).
  To make this concrete, consider the error in angle measurements that would arise from a small perturbation of a robot's local coordinate system: a *symmetric distortion* of some reference coordinate system is a continuous bijection $\mu : [0, 2\pi]) \to [0, 2\pi)$ with $\mu(\theta+\pi) = \mu(\theta)+\pi$, for all $\theta \in [0, \pi)$. Such a distortion has *skew* bounded by $\lambda < 1$ if $(1-\lambda)\xi \leq \mu(\theta+\xi) - \mu(\theta) \leq (1+\lambda)\xi$, for all $\xi \in (0, \pi/2)$.

  Observe that a symmetric distortion with skew 0 is just a rotation or reflection of the local coordinate system. That a permissible distortion is a continuous bijection seems natural; that it should also be symmetric is also realistic in our context, since otherwise it is possible to construct (as above) a configuration with robots located at the vertices of a regular polygon with edge length V, that fails to converge.

  The KKNPS algorithm can be modified to tolerate a symmetric distortion of its local coordinate system with small skew. The intuition here is to reduce the safe region of robot $\mathbb{Y}$ with respect to a distant neighbour $\mathbb{X}$ by making it instead the intersection of the safe regions that would be associated with all possible true locations of $\mathbb{X}$.
- Even linear relative error (error that grows linearly with the distance travelled), measured as the deviation of the actual destination from the trajectory to the intended destination, cannot be tolerated by any algorithm that solves POINT CONVERGENCE (see Fig. 18). even in the fully synchronous model.

  However, the KKNPS algorithm can be modified to tolerate relative motion error that grows quadratically (specifically error is $O(d^2/V)$, where d is the distance travelled). To see this, recall that the KKNPS algorithm chooses as its destination point the midpoint of the region formed by the intersection of the safe regions of a robot's extreme distant neighbours. If the angle formed by these neighbours is bounded away from π then there is fixed faction of the intended trajectory that, even when followed with quadratic error, will remain within the intersection of the safe regions.

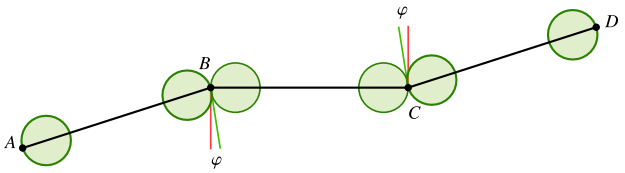

Fig. 18 If the relative motion error is greater than $\tan\varphi$ then B can move to the left of the normal to BC (red) and C can move to the right, resulting in separation if $|BC| = V$ (colour figure online)

- The KKNPS algorithm can clearly tolerate premature stop failures, provided these do not accumulate in a way that violates the activation fairness condition. In fact, a single crash (or "fail-stop") fault, where a robot crashes and remains stationary thereafter, can also be tolerated. In this case the remaining robots converge to the location of the crashed robot.

7.3 Visibility

The KKNPS algorithm can be modified without difficulty to work correctly even if the visibility radius V may differ for different robots, provided that (i) the initial mutual visibility graph is connected, and (ii) individual (unknown) visibility radii could differ by at most some small (known) constant factor.

Recall that the visibility region of robots was defined to be a closed set. However, the KKNPS algorithm works correctly even if the visibility region is open (i.e., to be visible a neighbouring robot must be at distance strictly less than V). To see this, it suffices to observe that in this case, $V_{\mathbb{Z}}$, the perceived distance to the furthest neighbour, is always strictly smaller than V.

Furthermore, we can modify the notion of visibility region to accommodate a more realistic *soft visibility threshold*. With this modification, there are two thresholds V and V^+; robots at distance less than or equal to V are guaranteed to be visible, those at distance greater than V^+ are guaranteed to be invisible, and those whose distance is between V and V^+ may or may not be visible. Since the KKNPS algorithm works correctly using any lower bound on the threshold V, and the distance to a robot's current furthest neighbour provides a lower bound on V^+, it suffices to use that distance scaled by V/V^+ as a conservative bound on V.

Finally, we note that in the event that the visibility threshold V exceeds the diameter of the initial configuration, it follows from the hull-diminishing property that all robots will remain mutually visible throughout the computation. Since our congregation argument continues to hold even under an ASYNC scheduler, we conclude that our 1-ASYNC algorithm will guarantee convergence in this case, *without any need for multiplicity detection*.

7.4 Further relaxations

7.4.1 More general initial conditions for guaranteeing convergence

It is worth noting that if the initial configuration is not connected the KKNPS algorithm guarantees that every connected subset of robots will converge to a single point. Moreover, noting that (i) all edges of the initial visibility graph are preserved, and (ii) a robot $\mathbb{X}$ cannot cross a visibility edge without becoming strongly connected to one of robots positioned at that edge's endpoints (and remaining connected thereafter), it follows that robots in an interior face of the initial visibility graph must ultimately converge to the same point as the robots defining that face. Thus, connectivity of the initial configuration is not a strictly necessary condition for the KKNPS algorithm to solve COHESIVE CONVERGENCE.

In contrast, it is not hard to see that if a robot $\mathbb{X}$ is positioned in the exterior face of some component C of the initial visibility graph, then $\mathbb{X}$ is not guaranteed to converge with the robots in C, even if $\mathbb{X}$ lies inside the convex hull of the robots in C.

7.4.2 Motion in three dimensional space

As noted in Observation 2, the KKNPS algorithm admits a very natural generalization for robots configured in three dimensions. We have also noted in passing how the arguments supporting the correctness of the KKNPS algorithm for robots moving in two dimensions can be extended to three dimensions. We revisit and expand on some of these arguments here.

Recall that the correctness of the KKNPS algorithm was proved, like that of its predecessors, by establishing two properties: (i) *visibility preservation*: the choice of safe region guarantees that all robot pairs that start or become (strongly) visible will remain mutually visible thereafter, and (ii) *incremental congregation*: the trajectories of robots following the algorithm convergence monotonically (measured in terms of the shrinking of the convex hull of the robot locations) to a single point.

We pointed out in Observation 6 that the visibility preservation property, established in Lemma 4, extends to three dimensions using a "trajectory flattening" argument. This argument involves showing that every engagement of a pair of robots $\mathbb{X}$ and $\mathbb{Y}$ can be replaced by a new valid engagement in which (i) all of the activation points in each activation cluster of one robot are co-planar with the trajectory of the other robot in its current active interval, and (ii) all distances between successive activation checkpoints remain unchanged.

Consider some engagement of robots $\mathbb{X}$ and $\mathbb{Y}$, and let $X_{j,1}, \ldots, X_{j,s_j}$, where $s_j \leq k$, denote the positions of robot $\mathbb{X}$ at the start of the activity intervals in its jth activation cluster, which occurs as robot $\mathbb{Y}$ is executing the last activity interval of its $(j-1)$st activation cluster (recall Fig. 14). Suppose that points $X_{j,1}, \ldots, X_{j,m}$ are all co-planar with points $Y_{j-1,t_{j-1}}$, $Y_{j,1}$ and $X_{j,1}$, where $1 \leq m < s_j$. and consider the point $X'_{j,m+1}$ formed by rotating the triangle formed by $Y_{j-1,t_{j-1}}$, $Y_{j,1}$ and $X_{j,m+1}$ about the line through points $Y_{j-1,t_{j-1}}$ and $Y_{j,1}$ until the triangle becomes co-planar with points $Y_{j-1,t_{j-1}}$, $Y_{j,1}$ and $X_{j,1}$. Since (i) $|X'_{j,m+1}Y| = |X_{j,m+1}Y|$ for all $Y \in \overline{Y_{j-1,t_{j-1}}Y_{j,1}}$ and (ii) $X'_{j,m+1}$ is co-planar with points $Y_{j-1,t_{j-1}}$, $Y_{j,1}$ and $X_{j,1}$, it suffices to observe that $X'_{j,m+1}$ must lie closer to the centre of $S^{V_{\mathbb{X}}/(8k)}_{X_{j,m}}$ than $X_{j,m+1}$ and hence $X'_{j,m+1} \in S^{V_{\mathbb{X}}/(8k)}_{X_{j,m}}$.

As noted in Observation 8 the incremental congregation property, established in Lemma 8, also extends to robots moving in three dimensions. To see this note (building on the proof of Lemma 8, where Figs. 16 and 17 should be interpreted as the projection of $CH_{t'}$, the convex hull of the robot positions at time t', onto a plane through A_H and O_H) that the pyramid with apex A_H cut off from $CH_{t'}$ by the plane with normal $\overline{A_H O_H}$ at distance x_d from A_H, has a surface area that exceeds its base area by an amount that is at least $2(d-z_d)$ times the perimeter of its base. Thus the surface area of the convex hull after $\Gamma_\delta(A_H)$ becomes empty is smaller than that of $CH_{t'}$ by an amount that exceeds some constant independent of ε, or the pyramid base has arbitrarily small perimeter. In the latter case, $CH_{t'}$ is contained in a cone with small solid angle at apex A_H, from which it follows directly that the radius of the convex hull after $\Gamma_\delta(A_H)$ becomes empty is at least some constant independent of ε less than r_H.

8 Impossibility of visibility preservation in the ASYNC scheduling model, assuming modest error in perception, but no error in motion

Let $\mathcal{A}$ be any control algorithm for $\mathcal{OBLOT}$ robots. In this section we will show that if $\mathcal{A}$ (i) is even slightly error-tolerant and (ii) solves POINT CONVERGENCE on all configurations of robots that form a regular polygon (even if this requires the aid of a benevolent scheduler), then we can associate with $\mathcal{A}$ an initial connected configuration of robots, together with an adversarial scheduler in the ASYNC model (in fact, the NESTA model) that produces a robot configuration with two or more linearly separable connected components in its associated visibility graph. Specifically, we consider algorithms, like the one developed in preceding sections, and like those that we might reasonably expect to be used in practice, that

tolerate the kind of error associated with *realistic robots*. Such robots have a perceptual apparatus that is only accurate up to (i) small relative error in distance measurements (the perceived distance $\tilde{d}$ to any neighbour of a robot is accurate only to within a constant $\delta > 0$ times the true distance d), and (ii) small relative errors in angle measurements of the kind that arise from symmetric distortions of local coordinate systems, with bounded skew (the perceived angle $\tilde{\theta}$ formed any two neighbours is accurate only to within some constant $0 < \lambda < 1$ times $\min\{\theta, \pi - \theta\}$, where $\theta \in [0, \pi]$ is the true angle). The control algorithm is free to assume a fixed visibility threshold (assumed to be 1), and that motion is rigid and free of error.

Assumption (i) allows us to specify an initial configuration of robots so that the true distance of neighbouring robots, in both this initial configuration and all subsequent configurations (up to the point of disconnection of the visibility graph), is sufficiently close to the visibility threshold that neighbours could be perceived as always being exactly at that threshold. Assumption (ii) allows us to conclude that, while the perceived sidedness of the observing robot with respect to the line joining any pair of its neighbours agrees with reality, there is sufficient uncertainty to guarantee that robots that are not yet co-linear with their neighbours must move. (Otherwise, as we will show, it would be possible to construct a local configuration that likewise has no associated forced move, but which could be replicated into a global configuration that, for this reason, fails to converge, even with a benevolent scheduler).

It follows that the COHESIVE CONVERGENCE task is not solvable in the ASYNC or NESTA model, for all arbitrarily large collections of realistic robots. This in turn implies that the POINT CONVERGENCE task is not solvable for realistic robots in the ASYNC model, using any hull-diminishing algorithm. In fact, even if an algorithm is only *weakly cohesive* (in the sense that connectivity of the visibility graph is preserved over time, even if mutual visibility between some robot pairs might be broken) then it cannot guarantee convergence on all initial configurations whose visibility graph is connected.

We conclude that the POINT CONVERGENCE task illustrates a separation in the power of bounded vs. unbounded asynchrony, for natural (hull-diminishing or even weakly cohesive), modestly error-tolerant algorithms. Note that, it seems unavoidable to insist on some measure of error tolerance in framing this result. In fact, a proof that something is impossible for error-free algorithms with rigid motion seems well out of reach.

Suppose then that algorithm $\mathcal{A}$ is error-tolerant, in the sense described above, and that it solves POINT CONVERGENCE on all connected robot configurations that form a regular polygon.

8.1 The initial configuration

We describe the initial configuration in terms of a parameter $\psi > 0$ that we will subsequently set to be sufficiently small. We take the visibility threshold V to be 1.

The initial robot configuration consists of three robots $\mathbb{X}_A$, $\mathbb{X}_C$ and $\mathbb{X}_B$, located at positions $A = (0, 0)$, $C = (-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$ and $B = (1, 0)$ respectively, and $n - 3$ additional robots, $\mathbb{X}_1, \mathbb{X}_2, \ldots, \mathbb{X}_{n-3}$, positioned at points $P_1, P_2, \ldots, P_{n-3}$ spaced at distance 1 along a discrete spiral tail starting with the edge AB. The turn angle between the chord $\overline{AP_{i-1}}$ and the segment $\overline{P_{i-1}P_i}$, taking $P_0 = B$, is fixed at ψ, for all $i \geq 1$. (Fig. 19 (left) illustrates such a configuration).

The total number of robots, n, is chosen to be sufficiently large that the angle between the chords $\overline{AP_0}$ and $\overline{AP_{n-3}}$ is about $3\pi/8$ (so the line through A and P_n bisects the angle $\angle(CAB)$). Of course, we need to argue that such an n exists: for this we observe that even though the angle γ_i between successive chords $\overline{AP_{i-1}}$ and $\overline{AP_i}$ decreases with i, the sum $\sum_i \gamma_i$ diverges. To see this, let d_i denote the length of the chord $\overline{AP_i}$. We know, from the cosine law, that $d_i^2 = d_{i-1}^2 + 1 - 2d_{i-1}\cos(\pi - \psi)$. Hence, when $\psi > 0$,

$$\begin{aligned} d_i - d_{i-1} &= \frac{1 + 2d_{i-1} - 2d_{i-1}(1 - \cos\psi)}{d_i + d_{i-1}} \\ &> \frac{1 + 2d_{i-1} - 2d_{i-1}(1 - \cos\psi)}{1 + 2d_{i-1}} \\ &= 1 - \frac{2d_{i-1}(1 - \cos\psi)}{1 + 2d_{i-1}} \geq 1 - \frac{2d_{i-1}(\psi^2/2)}{1 + 2d_{i-1}} \\ &> 1 - \psi^2/2. \end{aligned}$$

Since $d_0 = 1$, and $d_i - d_{i-1} \leq 1$ (by the triangle inequality), it follows by induction on i that $1 + i(1 - \psi^2/2) < d_i \leq 1 + i$, for $0 \leq i \leq n - 3$. Since the scheduler maintains the distance from point A to the location of robot $\mathbb{X}_i$ close to d_i, choosing ψ small enough guarantees that the separation between successive robots remains close to 1. Hence each robot on the chain remains visible only to its initial neighbours.

It is easy to confirm that the perpendicular distance from P_i to the line through A and P_{i-1} equals both $\sin\psi$ and $d_i \sin\gamma_i$. Hence $\gamma_i \geq \sin\gamma_i = \frac{\sin\psi}{d_i} \geq \frac{\sin\psi}{i+1}$. It follows that $\sum_{i=1}^{n-3} \gamma_i \geq \sin\psi \ln\frac{n-3}{e}$. Thus it will suffice to choose n large enough that $\sin\psi \ln\frac{n-3}{e} \geq 3\pi/8$, i.e., $n \geq 3 + e^{\frac{3\pi}{8\sin\psi}+1}$.

8.2 Adversarial scheduler strategy

The strategy of the adversarial scheduler is to first activate the robot $\mathbb{X}_A$ forcing it (as we will argue) to plan a move to a point A' some non-zero distance ζ away from A into the sector CAB. By the symmetry of the local configuration, we

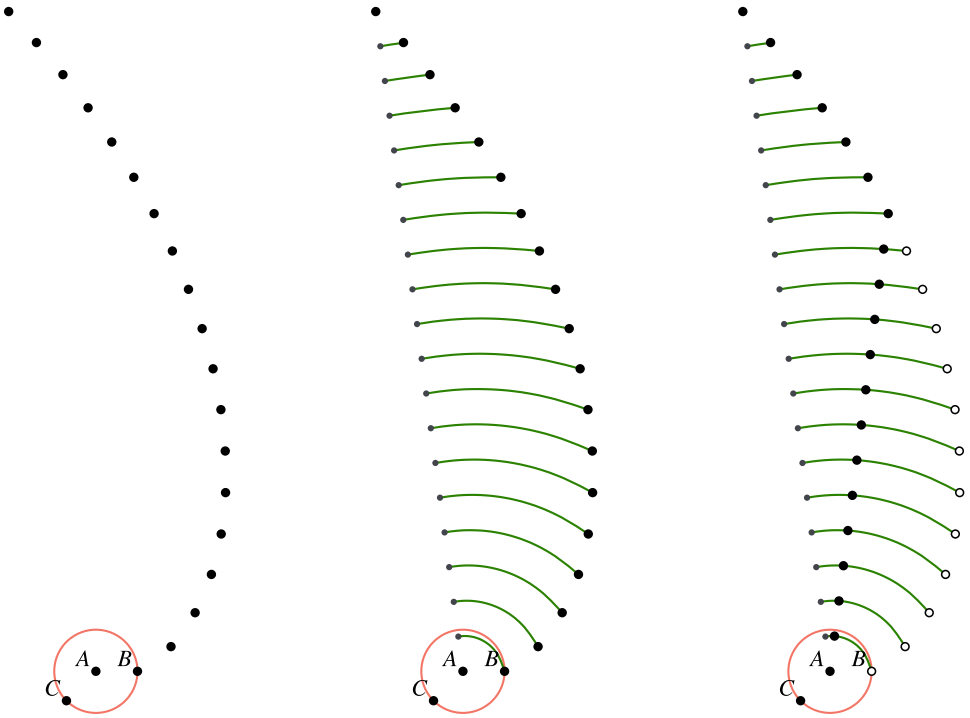

Fig. 19 Left: initial robot configuration; Center: approximate trajectories to final configuration; Right: typical intermediate configuration

can assume that $|A'C| \leq |A'B|$; for ease of description, we will assume that these distances are equal.

Next, and before the robot $\mathbb{X}_A$ actually begins its move, the scheduler begins a long sequence of activations of the robots $\mathbb{X}_0, \ldots, \mathbb{X}_{n-3}$ (where $\mathbb{X}_0 = \mathbb{X}_B$), with the goal of moving, for each i, $1 \leq i \leq n-3$, in succession, all of the robots $\mathbb{X}_0, \mathbb{X}_1, \ldots, \mathbb{X}_{i-1}$ onto (or close to) the chord AP_i, without changing their initial distance from A by more than an amount that is $\Theta(\psi^2)$. If successful in this task then, by choosing ψ to be sufficiently small, all robots will follow trajectories that approximate circular arcs, centered at A, from their initial position to their final position on the chord $\overline{AP_n}$ (see Fig. 19 (center) and (right)). Furthermore, if ψ is chosen to be sufficiently small relative to ζ, the scheduler will have succeeded in breaking the visibility between the robots $\mathbb{X}_A$ and $\mathbb{X}_B$.

The argument that such an activation sequence exists involves two things: (i) showing that specified robots must move, when activated, and (ii) showing that the motion must keep their distance from A the same, up to an additive factor that accumulates to something that is $O(\psi^2)$.

The space between chords AP_{i-1} and AP_i defines what we call the ith *sliver* (see Fig. 20). The transitions of robots from one chord to the next, flattening the associated sliver, involves the (essential) collapse of a succession of thin triangles, each of which involves the relocation of a robot to become essentially co-linear with its neighbours at almost distance one. By "essentially co-linear" we mean that the angle formed by the neighbours is in $(\pi - \varphi_0, \pi]$, where angle φ_0 is chosen to be sufficiently small (relative to ψ and $1/n^2$). This slight slack permits termination of each flattening step and ensures that when the flattening of the ith sliver is complete the sum of all turn angles on the chain formed by $\mathbb{X}_0, \ldots, \mathbb{X}_i$ is at most $i\varphi_0$ and hence all of the robots lie within distance $i^2\varphi_0/2$ of the chord AP_i. Hence, when all slivers have been flattened, robot $\mathbb{X}_B$ lies within distance $n^2\varphi_0/2$ of the chord AP_{n-3}.

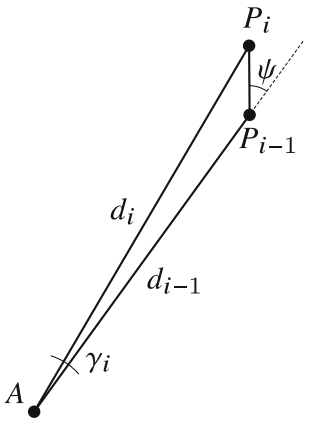

Fig. 20 Initial sliver

8.2.1 Forced motion

Imagine that a triple of robots is in a local configuration like that illustrated in Fig. 21. Under what conditions can we assume that the robot at position Q must move if activated (which only happens if the perceived turn angle is at least φ_0)? Our assumption about error in distance perception is such that if the distances between Q and its two neighbours are both in the range $(1-\delta, 1]$, then they could both be perceived as being 1. Clearly, this would preclude algorithm $\mathcal{A}$ from choosing not to move the robot at Q (when activated) if in addition the angle φ could be perceived to be of the form $2\pi/K$, for some integer K; otherwise the algorithm would fail to converge if started from a configuration with robots located at the vertices of a regular K-gon with unit length sides, contradicting our assumption. Thus, if the error in angle perception is such that the angle φ could be perceived to have this special form, some motion of the robot at Q is forced.

Our assumption about error in angle perception is such that the true turn angle φ could be perceived as anything in the range $[\varphi(1-\lambda), \varphi(1+\lambda)]$, for some positive constant $\lambda < 1$. Since, as we will see, the turn angle in all configurations to be collapsed, is at most ψ, it will suffice to choose $\psi \leq \frac{2\pi}{\lceil 1/\lambda \rceil}$ to guarantee that an angle of special form exists in the range $[\varphi(1-\lambda), \varphi]$. This follows because, when $K \geq \lceil 1/\lambda \rceil$, $\frac{2\pi}{K} \leq \varphi \leq \frac{2\pi}{K-1}$ implies $(1-\lambda)\varphi \leq \frac{2\pi}{K}$.

Observation 9 *It is worth noting that if integer M is greater than $\frac{4\pi}{\lambda\varphi_0}$, then there must exist angles $\frac{2\pi i}{M}$ and $\frac{2\pi(i+1)}{M}$ both in the range $[\varphi(1-\lambda), \varphi]$. This means that if the algorithm chooses not to move the robot at point Q (when activated) when the turn angle is perceived to be $\frac{2\pi i}{M}$ then it must move the robot at point Q (when activated) when the turn angle is perceived to be $\frac{2\pi(i+1)}{M}$. Otherwise, we could construct another polygonal placement with $2M$ edges (forming what we call a* semi-regular *polygon) that alternate a $\frac{2\pi(i+1)}{M}$ counter-clockwise turn with a $\frac{2\pi i}{M}$ clockwise turn, from which the algorithm would fail to move, and hence fail to converge. Thus our assumption "(ii) if $\mathcal{A}$ solves* POINT CONVERGENCE *on all configurations of robots that form a regular polygon (even if this requires the aid of a benevolent scheduler)" can be replaced by the much weaker assumption "(ii) if $\mathcal{A}$ solves* POINT CONVERGENCE *on some sufficiently large configuration of robots that forms a semi-regular polygon (even if this requires the aid of a benevolent scheduler)"*

8.2.2 Collapsing thin triangles

It is not enough to know that motion can be forced by an adversary. We also need to demonstrate that (i) motion is *productive*, in the sense that it eventually leads to essential co-linearity [16] with neighbours, and (ii) motion preserves distance to point A, up to some small additive error, proportional to angle φ and the distance moved.

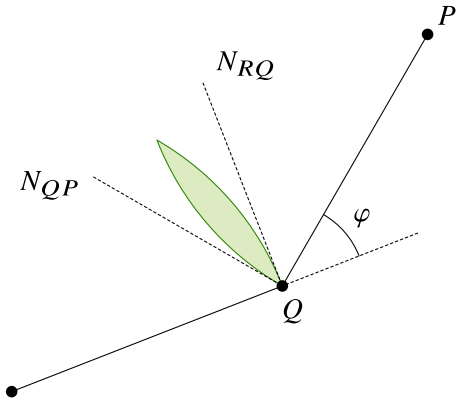

Fig. 21 Robot located at Q may be forced to move

Here again, it helps to consider a triple of robots is in a configuration like that illustrated in Fig. 21, where the distances $|RQ|$ and $|QP|$ are both perceived as 1 by the robot at Q. We know that the robot at location Q (when activated) is forced to move, and its intended destination must lie within the green lens; otherwise, such a move would result in a disconnected visibility graph if this were the full initial configuration.

Any motion only leads to a configuration with smaller turn angle φ. (This is true even if the motion takes Q beyond the midpoint of the lens). Since the new turn angle φ' could continue to be perceived as being at least φ_0 as long as $\varphi' > \varphi_0/(1+\lambda)$, it follows that repeated activation of the same robot will keep it inside the lens and eventually bring it to a position Q' of essential co-linearity with its neighbours (i.e with a turn angle at most φ_0), provided that the distances to the neighbouring robots continues to be perceived at 1.

Any point Q' of co-linearity within the lens has distance ℓ at most $2\sin(\varphi/4)$, which is at most $\varphi/2$, from Q. Furthermore, the distance from Q' to both of the normals N_{RQ} and N_{QP} is at most $\ell \sin\varphi$. From this it follows that the distance from Q' to any point D at or beyond R on the ray from Q through R satisfies $|DQ| - \varphi^2/2 \leq |DQ| - \ell\sin\varphi < |DQ'| < |DQ|$. In particular, once the robot at Q has reached a position of essential co-linearity with its neighbours, its distance from point A, the base point of our global configuration, has not changed by more than $\varphi^2/2$.

[16] To simplify the argument, we ignore the fact that "essential co-linearity" is not exactly the same as co-linearity. This is justified by the fact that, as we will demonstrate, robots can be moved to become arbitrarily close to co-linear, and the extra work involved in carrying insignificant error terms would not make the argument stronger or more illuminating in any measurable way.

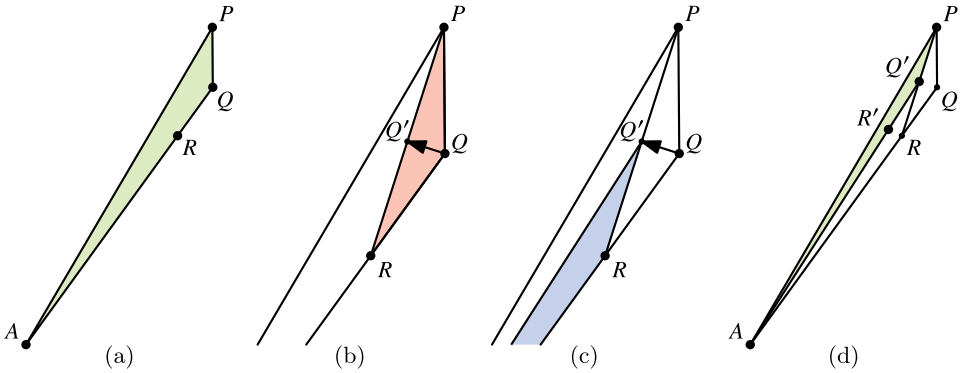

Fig. 22 Recursive collapse of a sliver

8.2.3 Flattening slivers

A sliver (see Fig. 22a, for a generic example) is iteratively flattened by first (i) moving the endpoint Q of the first chord until it is (essentially) co-linear with its neighbours R and P, collapsing the associated triangle (see Fig. 22b), as described in the preceding subsection, then (ii) recursively flattening the newly created sliver ARQ' (see Fig. 22c), and finally (iii) continuing to flatten the semi-flattened sliver $AQ'P$ (see Fig. 22d).

It remains to argue that the change in the distance between the location of robot $\mathbb{X}_j$ and A, even though it may accumulate through a long sequence of triangle collapses, does not sum to more than $4\psi^2$. To see this we observe that:

(i) all slivers formed by successive chords in the initial placement have turn angle ψ;
(ii) the successively narrower slivers (the green slivers in Fig. 22a, d) that arise in flattening any initial sliver have turn angles that decrease by at least a factor of two (since $(\pi - \angle AQ'P) < \angle Q'RQ = (\pi - \angle RQP)/2$)
(iii) the sliver that arises after the first triangle collapse (the blue sliver in Fig. 22c) has a turn angle $(\pi - \angle ARQ')$ that is exactly one half of that its parent sliver $(\pi - \angle RQP)$; and more generally the slivers that arise in the tth level of recursive sliver flattening have a turn angle that is a fraction $1/2^t$ of that of its parent sliver.

It follows that the change in distance from A associated with the motion of robot $\mathbb{X}_j$ in the course of moving it from its location on chord AP_i to its location on chord AP_{i+1} is at most $\sum_{t\geq 0}(\frac{\psi}{2^{i-j+t}})^2/2 < \frac{\psi^2}{2^{i-j}}$. Hence the total deviation of $\mathbb{X}_j$ from its initial distance d_j is at most $\sum_{i\geq j}\frac{\psi^2}{2^{i-j}} < 2\psi^2$.

In conclusion, we have established the following:

Theorem 2 *The* Point Convergence *task is not solvable in the* Async *or* NestA *scheduling models using natural (hull diminishing) and modestly error-tolerant algorithms.*

Together Theorems 1 and 2 show that:

Corollary 1 *The* Point Convergence *task provides a separation in the power of bounded and unbounded asynchrony, for realistic robots.*

Observation 10 *Building on Observation 9, it follows that the impossibility result of this section holds even if the number of robots n is known to the algorithm, provided (i) n is sufficiently large and (ii) there is a positive constant ζ_0 such that the distance ζ is at least ζ_0, for all n.*

9 Summary and conclusion

We have studied the well-known Point Convergence task for autonomous mobile robots with bounded visibility. Connectivity, realized through visibility, plays a fundamental role in all known algorithms for Point Convergence in this setting: it is typically assumed that the visibility graph of the initial robot configuration is connected, and a central algorithm design constraint is the preservation of visibility between pairs of robots. Indeed all previous algorithms solve the apparently more restricted task, which we call Cohesive Convergence, that requires the visibility between all initially mutually visible robot pairs to be preserved indefinitely.

We have presented a new algorithm (KKNPS) for Cohesive Convergence designed for robots operating within the k-Async scheduling model, for any constant k, a significantly more inclusive scheduling environment than that assumed by its predecessors. The KKNPS algorithm need not have exact knowledge of k; an upper bound suffices. In addition, the KKNPS algorithm makes comparatively weak assumptions about robot capabilities, including non-rigid movements and no knowledge of the visibility threshold V, which itself need not be sharp. Another significant distinguishing feature is the KKNPS algorithm's tolerance for limited imprecision in the distances and angles perceived by the robots, and limited relative error in the realization of

intended motions. Although formulated for robots moving in two dimensions, the KKNPS algorithm has a very natural extension to three dimensions, with somewhat more involved correctness proofs. As it happens, the KKNPS algorithm, without modification, also solves POINT CONVERGENCE in the fully asynchronous (ASYNC) scheduling model, when visibility is unbounded, without depending on the ability to detect multiplicities in robot locations.

We also establish a complementary result that COHESIVE CONVERGENCE cannot be solved in the fully asynchronous (ASYNC) scheduling environment, assuming that robots are subject to very limited imprecision in measurements, even assuming rigid motion and exact knowledge of a sharp visibility threshold V. Hence, COHESIVE CONVERGENCE demonstrates a separation between the power of autonomous robots operating in a scheduling models with arbitrarily high but bounded asynchrony (k-ASYNC), and those operating in the presence of unbounded asynchrony (ASYNC). As a special case, this also provides a positive answer to a long-standing open question: whether robots operating in a semi-synchronous (SSYNC) scheduling environment are strictly more powerful than those operating in a fully asynchronous (ASYNC) scheduling environment.

Our results leave open several natural questions. Among these we note:

1. Are there problems that serve to distinguish the power of robots operating in the k-ASYNC and SSYNC scheduling environments?
2. Is (COHESIVE) POINT CONVERGENCE solvable in the fully asynchronous (ASYNC) scheduling environment if there is no error in perception or motion, or if error is restricted to some arbitrarily small relative quantity arising from motion alone?
3. From our investigation and from the literature, it turns out to be natural to consider COHESIVE CONVERGENCE rather than simply POINT CONVERGENCE. However, it is not clear whether hull-diminishing or weakly cohesive algorithms, as natural as they may be, can actually be avoided. In general, understanding whether the resolution of POINT CONVERGENCE necessarily solves COHESIVE CONVERGENCE as well remains an open question.

As future work, it would be interesting to extend our results to consider robots with non-zero extent, as in the *fat* robots studied in [16, 45, 46]. Other natural constraints worth investigating include occlusion, as in [4, 7], and collision avoidance, as in [43].

Acknowledgements We would like to acknowledge the stimulating research environment provided by the Bertinoro Workshop on Distributed Geometric Algorithms, held in August 2019, where the seeds of this paper were planted. In particular, we gratefully acknowledge the discussions with Pierre Fraigniaud at the beginning of this work. This research project has been supported in part by the Natural Sciences and Engineering Research Council (Canada) under Discovery Grants A3583 and A2415, and by the Italian National Group for Scientific Computation GNCS-INdAM.

Author Contributions These authors contributed equally to this work.

Funding Open access funding provided by Università di Pisa within the CRUI-CARE Agreement.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Kirkpatrick, D., Kostitsyna, I., Navarra, A., Prencipe, G., Santoro, N.: Separating bounded and unbounded asynchrony for autonomous robots: point convergence with limited visibility. In: 40th ACM Symposium on Principles of Distributed Computing (PODC), pp. 9–19 (2021)
2. Flocchini, P., Prencipe, G., Santoro, N.: Distributed Computing by Oblivious Mobile Robots. Morgan & Claypool, Williston (2012)
3. Braun, M., Castenow, J., Heide, F.M.: Local gathering of mobile robots in three dimensions. CoRR **abs/2005.07495** (2020)
4. Bhagat, S., Chaudhuri, S.G., Mukhopadyaya, K.: Gathering of opaque robots in 3D space. In: 19th International ACM Conference on Distributed Computing and Networking (ICDCN), pp. 1–10 (2018)
5. Yamauchi, Y., Uehara, T., Yamashita, M.: Brief announcement: pattern formation problem for synchronous mobile robots in the three dimensional Euclidean space. In: Giakkoupis, G. (ed.) Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC), pp. 447–449 (2016). https://doi.org/10.1145/2933057.2933063
6. Yamauchi, Y., Uehara, T., Kijima, S., Yamashita, M.: Plane formation by synchronous mobile robots in the three-dimensional Euclidean space. J. ACM **64**(3), 16–11643 (2017)
7. Bose, K., Kundu, M.K., Adhikary, R., Sau, B.: Arbitrary pattern formation by asynchronous opaque robots with lights. Theor. Comput. Sci. **849**, 138–158 (2021)
8. Cicerone, S., Di Stefano, G., Navarra, A.: Asynchronous arbitrary pattern formation: the effects of a rigorous approach. Distrib. Comput. **32**(2), 91–132 (2019)
9. Cicerone, S., Di Stefano, G., Navarra, A.: Solving the pattern formation by mobile robots with chirality. IEEE Access **9**, 88177–88204 (2021)
10. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Arbitrary pattern formation by asynchronous oblivious robots. Theor. Comput. Sci. **407**(1–3), 412–447 (2008)

11. Kasuya, M., Ito, N., Inuzuka, N., Wada, K.: A pattern formation algorithm for a set of autonomous distributed robots with agreement on orientation along one axis. Syst. Comput. Japan **37**(10), 89–100 (2006)
12. Agmon, N., Peleg, D.: Fault-tolerant gathering algorithms for autonomous mobile robots. SIAM J. Comput. **36**, 56–82 (2006)
13. Bouzid, Z., Das, S., Tixeuil, S.: Gathering of mobile robots tolerating multiple crash faults. In: 33rd International Conference on Distributed Computing Systems (ICDCS), pp. 337–346 (2013)
14. Cicerone, S., Di Stefano, G., Navarra, A.: Gathering of robots on meeting-points: feasibility and optimal resolution algorithms. Distrib. Comput. **31**(1), 1–50 (2018)
15. Cieliebak, M., Flocchini, P., Prencipe, G., Santoro, N.: Distributed computing by mobile robots: gathering. SIAM J. Comput. **41**(4), 829–879 (2012)
16. Czyzowicz, J., Gasieniec, L., Pelc, A.: Gathering few fat mobile robots in the plane. Theor. Comput. Sci. **410**(6–7), 481–499 (2009)
17. Izumi, T., Souissi, S., Katayama, Y., Inuzuka, N., Defago, X., Wada, K., Yamashita, M.: The gathering problem for two oblivious robots with unreliable compasses. SIAM J. Comput. **41**(1), 26–46 (2012)
18. Izumi, T., Katayama, Y., Inuzuka, N., Wada, K.: Gathering autonomous mobile robots with dynamic compasses: an optimal result. In: 21st International Symposium on Distributed Computing (DISC), pp. 298–312 (2007)
19. Lin, J., Morse, A.S., Anderson, B.D.O.: The multi-agent rendezvous problem. Part 2: the asynchronous case. SIAM J. Control Optim. **46**(6), 2120–2147 (2007)
20. Prencipe, G.: Impossibility of gathering by a set of autonomous mobile robots. Theor. Comput. Sci. **384**(2–3), 222–231 (2007)
21. Ando, H., Oasa, Y., Suzuki, I., Yamashita, M.: Distributed memoryless point convergence algorithm for mobile robots with limited visibility. IEEE Trans. Robot. Autom. **15**(5), 818–828 (1999)
22. Bar-Lev, A., Cohen, R.: Convergence of asynchronous autonomous mobile robots with inaccurate sensors. In: Algorithms and Experiments for Wireless Networks (ALGOSENSORS). Springer, Berlin, Heidelberg (2020) (under review)
23. Bouzid, Z., Potop-Butucaru, M.G., Tixeuil, S.: Byzantine convergence in robot networks: the price of asynchrony. In: Proceedings of 13th International Conference Principles of Distributed Systems (OPODIS), pp. 54–70 (2009)
24. Cord-Landwehr, A., Degener, B., Fischer, M., Hüllmann, M., Kempkes, B., Klaas, A., Kling, P., Kurras, S., Märtens, M., Heide, F., Raupach, C., Swierkot, K., Warner, D., Weddemann, C., Wonisch, D.: A new approach for analyzing convergence algorithms for mobile robots. In: Automata, Languages and Programming (ICALP). Lecture Notes in Computer Science, vol. 6756, pp. 650–661. Springer (2011)
25. Cohen, R., Peleg, D.: Convergence properties of the gravitational algorithm in asynchronous robot systems. SIAM J. Comput. **34**(6), 1516–1528 (2005)
26. Cohen, R., Peleg, D.: Convergence of autonomous mobile robots with inaccurate sensors and movements. SIAM J. Comput. **38**(1), 276–302 (2008)
27. Katreniak, B.: Convergence with limited visibility by asynchronous mobile robots. In: Structural Information and Communication Complexity (SIROCCO). Lecture Notes in Computer Science, vol. 6796, pp. 125–137. Springer (2011)
28. Pattanayak, D., Mondal, K., Mandal, P.S., Schmid, S.: Convergence of even simpler robots without position information. In: El Abbadi, A., Garbinato, B. (eds.) Networked Systems (NETYS). Lecture Notes in Computer Science, vol. 10299, pp. 69–85. Springer (2017)
29. Pattanayak, D., Mondal, K., Mandal, P.S., Schmid, S.: Area convergence of monoculus robots with additional capabilities. Comput. J. (COMPJ) (2021). https://doi.org/10.1093/comjnl/bxaa182
30. Yamamoto, K., Izumi, T., Katayama, Y., Inuzuka, N., Wada, K.: The optimal tolerance of uniform observation error for mobile robot convergence. Theor. Comput. Sci. **444**, 77–86 (2012)
31. Degener, B., Kempkes, B., Langner, T., Heide, F.M., Pietrzyk, P., Wattenhofer, R.: A tight runtime bound for synchronous gathering of autonomous robots with limited visibility. In: Proceedings of the 23rd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pp. 139–148 (2011)
32. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous robots with limited visibility. Theor. Comput. Sci. **337**(1–3), 147–168 (2005)
33. Souissi, S., Défago, X., Yamashita, M.: Using eventually consistent compasses to gather memory-less mobile robots with limited visibility. ACM Trans. Auton. Adapt. Syst. **4**(1), 1–27 (2009)
34. Yamauchi, Y., Yamashita, M.: Pattern formation by mobile robots with limited visibility. In: 20th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2013). Lecture Notes in Computer Science, vol. 8179, pp. 201–212. Springer. https://doi.org/10.1007/978-3-319-03578-9_17 (2013)
35. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: formation of geometric patterns. SIAM J. Comput. **28**(4), 1347–1363 (1999)
36. Yamashita, M., Suzuki, I.: Characterizing geometric patterns formable by oblivious anonymous mobile robots. Theor. Comput. Sci. **411**(26–28), 2433–2453 (2010)
37. Yamauchi, Y.: Symmetry of anonymous robots. Chapter 8 of: P. Flocchini, G. Prencipe, N. Santoro (eds.), Distributed Computing by Mobile Entities. Springer (2019)
38. Flocchini, P., Prencipe, G., Santoro, N., Viglietta, G.: Distributed computing by mobile robots: uniform circle formation. Distrib. Comput. **30**(6), 413–457 (2017)
39. Bhagat, S., Chaudhuri, S.G., Mukhopadyaya, K.: Fault-tolerant gathering of asynchronous oblivious mobile robots under one-axis agreement. J. Discrete Algorithms **36**, 50–62 (2016)
40. Défago, X., Potop-Butucaru, M., Parvédy, P.R.: Self-stabilizing gathering of mobile robots under crash or byzantine faults. Distrib. Comput. **33**(5), 393–421 (2020)
41. Cicerone, S., Di Stefano, G., Navarra, A.: Embedded pattern formation by asynchronous robots without chirality. Distrib. Comput. **32**(4), 291–315 (2019)
42. Das, S., Flocchini, P., Santoro, N., Yamashita, M.: Forming sequence of geometric patterns with oblivious mobile robots. Distrib. Comput. **28**, 131–145 (2015). https://doi.org/10.1007/s00446-014-0220-9
43. Pagli, L., Prencipe, G., Viglietta, G.: Getting close without touching: near-gathering for autonomous mobile robots. Distrib. Comput. **28**(5), 333–349 (2015)
44. Flocchini, P., Prencipe, G., Santoro, N.: Moving and computing models: robots. In: Distributed Computing by Mobile Entities, Current Research in Moving and Computing. Lecture Notes in Computer Science, Springer, vol. 11340, pp. 3–14 (2019). https://doi.org/10.1007/978-3-030-11072-7_1
45. Agathangelou, C., Georgiou, C., Mavronicolas, M.: A distributed algorithm for gathering many fat mobile robots in the plane. In: ACM Symposium on Principles of Distributed Computing (PODC), pp. 250–259 (2013)
46. Chaudhuri, S.G., Mukhopadhyaya, K.: Leader election and gathering for asynchronous fat robots without common chirality. J. Discrete Algorithms **33**, 171–192 (2015)