Marco Dorigo Mauro Birattari
Christian Blum Maurice Clerc
Thomas Stützle Alan F.T. Winfield (Eds.)

LNCS 5217

Ant Colony Optimization and Swarm Intelligence

6th International Conference, ANTS 2008
Brussels, Belgium, September 2008
Proceedings

Springer

Gathering Multiple Robotic Agents with Crude Distance Sensing Capabilities

Noam Gordon, Yotam Elor, and Alfred M. Bruckstein

Center for Intelligent Systems, CS Department
Technion — Israel Institute of Technology, Haifa, Israel
{ngordon,yotame,freddy}@cs.technion.ac.il

Abstract. In this follow-up to an ANTS 2004 paper we continue to investigate the problem of gathering a swarm of multiple robotic agents on the plane using very limited local sensing capabilities. In our previous work, we assumed that the agents cannot measure their distance to neighboring agents at all. In this paper, we consider a crude range-limited sensing capability that can only tell if neighboring agents are either *near* or *far*. We introduce two new variants of our previously proposed algorithm that utilize this capability. We prove the correctness of our algorithms, and show that the newly added capability can improve the performance of the algorithm significantly.

1 Introduction

The *gathering problem* is roughly defined as the problem of gathering multiple agents on the plane into a point or a small region, within finite or finite expected time. In some variants, it is sometimes also referred to as the problem of *point formation*, *convergence* or *rendezvous*. In the emerging field of theoretical swarm-robotic research, the gathering problem has been given increasing attention in recent years. This follows a general increase in interest in swarm robotics, and, in particular, what we feel as an increasing urge to attain more substantial theoretical backing to this field which has been initially mostly experimental. Being such a fundamental problem, a basis to many formation and consensus problems, it is an ideal setting for theoretical exploration of swarm robotics.

Several theoretical works on this subject exist. Current approaches include agreement on a meeting point with some unique geometrical property, assuming unlimited visibility [1,2,3,4]; using a common compass [5,6]; cyclic pursuit [7,8,9]; and others [10,11,12,13,14]. Sugihara et al. suggested a simple way to fill a convex shape, which is also useful for gathering [15].

These methods rely on strong assumptions about the agents: Some rely on labeling (e.g., pursuit), some on common orientation, and many on infinite-range visibility. Nearly all works rely on the agents' ability to measure their mutual distances. We focus on the problem under the *ant-robotic* paradigm, which assumes anonymous, homogeneous, memoryless agents lacking common knowledge and communication capabilities, and having only limited local sensing. In previous works [16,17] we proposed gathering algorithms which do not

M. Dorigo et al. (Eds.): ANTS 2008, LNCS 5217, pp. 72–83, 2008.

utilize distance sensing at all. To our best knowledge, the gathering problem under the ant-robotic model and without sensing distances has not been considered elsewhere.

The initial inspiration and motivation for our work came from experiments with real robots in our lab [18], made from LEGO parts and very simple sensors, which are range-limited and do not provide usable distance measurements.

The algorithm proposed in [16] was validated in simulations which showed that the agents always converge into a small dense cluster. The key property which ensured the convergence, was that the algorithm maintains mutual visibility. However, since the agents were not aware of their mutual distances, their movement was quite conservative, in order to maintain visibility. As a result, and as evidenced by our simulations, the convergence rate was slow. A formal correctness proof of this algorithm currently remains an open problem. In [17] we proposed a randomized variant of that algorithm, which we were able to prove, yet its behavior in simulations was similar with regard to the convergence rate.

In this work we investigate whether adding a crude distance sensing capability to the agents can help them gather more efficiently. We feel that this added ability, done minimally, does not violate the ant-robotic paradigm. It is plausible that robots will be able to tell *near* from *far* at the least. That's exactly what we give them, and the answer to our question is clearly affirmative.

In Sect. 2 we provide the basic definitions and the system model. In Sect. 3 we present and prove the termination of the main proposed algorithm. In Sect. 4 we present a variant of the model and the algorithm, which adds the capability of collapsing into nearby agents. We present and discuss simulations of the algorithms in Sect. 5, and conclude in Sect. 6. Most proofs were omitted due to space constraints, and will be published in a forthcoming paper.

2 The System Model

2.1 Basic Definitions

The *world* consists of the infinite plane $\mathbb{R}^2$ and n point *agents* living in it. We adapt Suzuki et al.'s convenient way of modeling a system of asynchronous agents [4], sometimes referred to as the *semi-synchronous* model: *Time* is a discrete series of *time steps* $t = 0, 1, \ldots$. In each time step, each agent may be either *active* or *inactive*, having no control over the random scheduling of its activity times. An active agent atomically senses its environment, performs calculations, and optionally moves instantly to another point within a distance σ (the *maximum step length*).

An agent is able to see other agents within distance V (the *visibility radius* or *range*). However, it cannot measure its exact *distance* from them. Rather, it can only tell if a visible agent is at either less or more than the *near-visibility* distance r. We assume that $3r < V$. There are no collisions. Several agents may occupy the same point. All agents are *memoryless*, *anonymous* (indistinguishable in their appearance) and *homogenous* (they lack any individuality or identity, and perform the same algorithm).

In what follows, we use the following definitions and notations:

- Denote a closed disc of radius R centered at a point p by $B_p(R)$;
- Denote the number of agents in the system by n.
- For an agent a, we also denote its position by a. The agent's position after moving (as described in the context) is denoted by a';
- Agents at a distance r or less are *nearby*. Otherwise, they are *far*;
- We term visibility between nearby agents *near-visibility*, and visibility between far agents *far-visibility*;
- Let b be an agent far-visible by agent a. The point on the line segment ab at a distance r from a is the *image* of b with regard to a. It is denoted by $\tilde{b}$.
- Denote by F the subset of the agents which have far-visible neighbors. Denote by $CH(F)$ the convex hull of the set F.
- For ease of notation we shall use F also to denote the *far-visibility graph*, whose nodes are the agents in F, and its edges are all *far-visibility edges*: $(a, b) \in F \iff r < \|a - b\| \leq V$;

2.2 Strong Asynchronicity

In Suzuki et al.'s original model, there are no assumptions regarding the activity schedule of the agents (except that no agent sleeps forever, i.e., each agent is active an infinite number of times). In our opinion, this proved to be too weak in the context of formation problems, as they could mostly achieve impossibility results in their works. In [4], they proved that there are no algorithms guaranteed to form shapes which are not purely symmetrical (e.g., perfect polygons). The reason was that the agents might not be able to break symmetries given certain schedules (e.g., if they happen to be synchronous). Perncipe et al. achieved similar impossibility results using similar reasoning under their more elaborate *totally asynchronous* model (See [19,20]), in which the agent activity cycles are neither atomic nor instantaneous. These results are very interesting in the context of computational swarm-robotic research, yet we feel that, in the research of real autonomous swarm-robotic systems, this weakness is somewhat artificial. Therefore, we revise the model by adding an assumption on the agent scheduling, striving as we can to make it as minimal and generic as possible, and reflect the natural asynchrony between autonomous robots. It is as follows:

Definition 2.1 (Strong Asynchronicity assumption). *There exists a constant $\varepsilon > 0$, such that for any subset S of the agents and in each time step t, the probability that S will be the set of active agents is at least ε.*

This assumption guarantees that any synchrony between robots will break in finite expected time, since there is always a probability of at least ε that it will break. More generally, the strong asynchronicity assumption makes us immune to adversarial schedules and allows us to prove the termination of our algorithms by construction, that is, if we show that there always exists a schedule that brings us to a goal configuration, then it is guaranteed that the algorithm will terminate in finite time. Let us formalize and generalize this idea in the following theorem.

Denote the state (or configuration) space by $\mathcal{C}$, and the subset of *goal configurations* by $G \subset \mathcal{C}$. The system is defined by $\mathcal{C}$, an initial configuration $c_0 \in \mathcal{C}$, and a Markovian transition function $\tau : \mathcal{C} \times \mathcal{C} \to [0, 1]$, which defines the transition probabilities between the states. Define a *reachable configuration* as one that it is possible to reach (directly or indirectly) from c_0. Denote the set of reachable configurations by $\mathcal{C}' \subseteq \mathcal{C}$.

Theorem 2.2. *A system will reach a goal configuration in finite expected time, if there exist constants M and $\varepsilon > 0$, such that for each reachable configuration $c \in \mathcal{C}'$, there exists a path in τ from c to some configuration in G, with at most M transitions, each having a probability of at least ε.*

Armed with this theorem, we can use a *constructive* approach in our proofs — we need to show that one can always construct a path from any configuration to the goal. The strong asynchronicity assumption provides the required lower bound ε on all transition probabilities.

3 The Algorithm

3.1 Definition

We mentioned above that the proposed algorithm is a variant of an algorithm presented in [16]. Let us begin with the original algorithm in Algorithm 1, which assumes no distance sensing ability. It is presented in first person, being performed by each agent from its own point of view.

Algorithm 1. Gathering with no distance sensing

1: **if** all of the visible agents lie within a wedge which spans less than half of my visibility disc **then**
2: move a step of length $\min(V \cos(\psi/2), V/2, \sigma)$ along the wedge's bisector, where ψ is the angle of the wedge.
3: **else**
4: do not move.

Beside the physical limitation σ, the step length is set so that visibility between the agents is maintained after they move, no matter what their mutual distances are. consequently, the algorithm is somewhat inefficient, as agents which are very close move (or do not move at all) much more conservatively than needed to maintain visibility. A central motivation in this work is to examine whether some crude knowledge of distance can improve the performance of the algorithm, hence the added capability to sense whether a visible agent is near or far. The new algorithm, Algorithm 2, is identical to the original, except that the agent ignores nearby agents, and the step length is different. For simplicity, we fix $\sigma = r$ in the remainder of this paper.

Algorithm 2. Gathering with crude near/far distance sensing

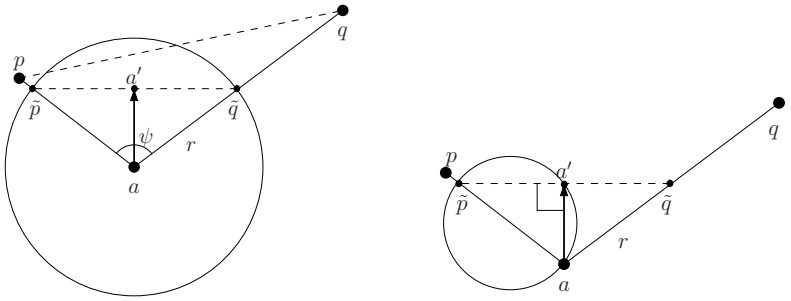
Let N be the set of all my neighbors at a distance between r and V.

1: **if** $N \neq \emptyset$ and all of the agents in N lie within a wedge which spans less than half of my visibility disc **then**
2: move a step of length $r\cos(\psi/2)$ along the wedge's bisector, where ψ is the angle of the wedge.
3: **else**
4: do not move.

In what follows, we call the agents that reside on the edges of the said wedge the *pulling* agents, as the agent seems to be "pulled" by these agents. We make the following simple yet important geometrical observations:

Remark 3.1. A moving agent's destination is the midpoint between the images of its pulling agents (See Fig. 1(a)).

Remark 3.2. Given the locations of agent a and the image of one of its pulling agents $\tilde{p}$, the set of all possible locations of its destination a' is a circle whose diameter is the line segment $a\tilde{p}$. This is due to Thales's classic theorem and the fact that a, a', and $\tilde{p}$ always form a right angle (See Fig. 1(b)). Take note of the two extreme cases: The case $a' = \tilde{p}$ corresponds to $\psi = 0$, where a is pulled only by $\tilde{p}$. The case $a' = a$ corresponds to $\psi = \pi$, where a doesn't move at all.



(a) a' is the midpoint between $\tilde{p}$ and $\tilde{q}$ and it's always inside the triangle apq.

(b) a' is always on the circle whose diameter is $a\tilde{p}$

Fig. 1. Geometric properties of agent movement

3.2 Proof

We now prove that Algorithm 2 indeed gathers all agents within an area of diameter r, in finite expected time. We use the constructive strategy suggested in Sect. 2.2, by showing that there always exists a bounded-time schedule that will either make the perimeter of $CH(F)$ shrink or new visibility links will be created. Chaining enough such sequences will generate a bounded-time schedule that shrinks $CH(F)$ into nothing. Then we apply this to Theorem 3.8.

Lemma 3.3. *The algorithm maintains visibility.*

Lemma 3.4. *Consider the following scenario:*

1. *An agent a moves (Others are not active);*
2. *Consequently, one or more nearby agents $b_i \notin F$ become far from a ($i = 1, \ldots, k,\ k > 0$);*
3. *Next, these agents become active and move.*

For each i,

a. *b_i' must be near a';*
b. *b_i' must see at least one of the pulling agents of agent a;*
c. *For each j, b_i' must be near b_j';*
d. *If, as a consequence of b_i's movement, some agent c near b_i becomes far from it, then c must be in F.*

Lemma 3.5. *A moving agent cannot move outside $CH(F)$.*

Let a_0 be the agent at a corner of the boundary of $CH(F)$, and let φ be the angle at that corner. Denote by A the isosceles triangle formed by the two boundary edges touching a_0 and a third line, such that the length of each of the triangle's edges touching a is $r \cos \frac{\varphi}{2}$.

Lemma 3.6. *For any agent in triangle A, if it becomes active and moves, it will move outside A.*

Lemmas 3.4, 3.5, and 3.6 are used to prove the following lemma, by constructing a schedule that empties the triangular corner A of F-agents, making $CH(F)$ shrink.

Lemma 3.7. *There exist constants $s^* > 0$ and $t^* \in \mathbb{N}$, such that, as long as $F \neq \emptyset$, there always exists an activity schedule that will decrease the perimeter of $CH(F)$ by at least s^* in at most t^* time steps, assuming that no new visibility links are created during that time.*

Theorem 3.8. *Given an initial configuration with a connected visibility graph, when performing Algorithm 2, the system will reach a static configuration of diameter r or less, in finite expected time.*

Proof. The initial visibility graph is globally connected and always remains so, due to Lemma 3.3, so the perimeter of $CH(F)$ is bounded. Lemma 3.7 always holds. Thus, we can always chain schedules constructed in Lemma 3.7 enough times so that the perimeter of $CH(F)$ becomes so small that it implies that its diameter is less than r. This, in turn, implies that $F = \emptyset$ by definition. The required number of these cycles is bounded, since each cycle decreases the perimeter by at least a minimal amount s^*, except possibly a bounded number of cycles during which new visibility links are created (since visibility is maintained, and there are $n(n-1)/2$ possible links in total). Each cycle's time is bounded

by t^*, so the resulting chained schedule is of bounded length (Let's denote the bound by M). We can construct such a schedule for any given configuration. The strong asynchronicity assumption gives a minimum probability ε for each transition. Therefore, Theorem 2.2 holds, and the system will reach a state where $F = \emptyset$ in finite expected time. It is straightforward to show that, together with the fact that the visibility graph is globally connected, this implies that all agents are mutually nearby, and so the diameter of the configuration is at most r. By definition of the algorithm, this configuration is static. □

4 A Variant — Gathering and Collapsing

When considering real robots, it is plausible to assume that in very close range $r \ll V$ the robots are able to sense each other better and perhaps communicate. Some works on formation (e.g., [21]) suggest that flocks of robots can move together as a whole, using some control hierarchy, where, for instance, a slave robot follows the movements of a master robot in the flock. In self-assembly and aggregation contexts, it is assumed that close robots can join in some rigid physical link, and effectively function and move as a single unit. In this section, we present a slightly modified model that abstracts these ideas, and a modified gathering algorithm for it.

4.1 Definitions

We use the same model and definitions as in Sect. 3.1, with the following modifications.

- We do *not* assume strong asynchronicity.
- An agent can move to the *exact* location of another *nearby* agent. Once it does so, it is permanently assimilated or *collapsed* into the other agent. From our point of view, the agent effectively disappears from the system.
- We regard the unification of nearby agents as a "low-level" action. Thus, we assume that in each time step, all collapses take place before movements. The order of collapses is arbitrary.

Algorithm 3 is quite similar to Algorithm 2, with the differences being a shorter step length (needed for our proof), and the collapsing into a nearby agent instead of just standing, when no far neighbors are visible.

4.2 Proof

The proof idea is somewhat similar to that of Algorithm 2, in the sense that we show that the convex hull (of *all* agents here, not just those in F) must shrink in finite time. Specifically, we show that the convex hull cannot expand and that some triangular corner of it must become empty once all agents wake up. We use the following definitions and notations:

Algorithm 3. Gathering with crude distance sensing and collapsing

Let N be the set of all my neighbors at a distance between r and V.

1: **if** $N \neq \emptyset$ and all of the agents in N lie within a wedge which spans less than half of my visibility disc **then**
2: move a step of length $\frac{1}{2}r\cos(\psi/2)$ along the wedge's bisector, where ψ is the angle of the wedge.
3: **else**
4: Collapse into my closest neighbor.

- Denote the convex hull of *all* agent locations by C;
- Let b be an agent visible by agent a. The point on the line segment ab at a distance $r/2$ from a is the *close image* of b with regard to a (Note the difference from the definition of the image of b in Sect. 2.1!).

Analogously to Remark 3.1 on Algorithm 2, we have the following remark:

Remark 4.1. A moving agent's destination is the midpoint between the close images of its pulling agents. This follows directly from the algorithm definition.

Remark 4.2. If an agent crosses a line as it moves, then at least one of its pulling agents must be across that line. This follows straightforwardly from the previous remark.

Lemma 4.3. *The algorithm maintains visibility.*

Lemma 4.4. *A moving agent cannot move outside C*

Let a_0 be the agent at a corner of the boundary of C, and let φ be the angle at that corner. Denote by A the isosceles triangle formed by the two boundary edges touching a_0 and a third line, such that the length of each of the triangle's edges touching a is $\frac{r}{2}\cos\frac{\varphi}{2}$. Note the differences from the definition of A in Sect. 3.2 — It is the corner of C (not $CH(F)$) and its dimensions are halved.

Lemma 4.5. *For any agent in triangle A, if it becomes active, it will either move outside A or collapse into another agent.*

Lemma 4.6. *An agent outside triangle A cannot enter it.*

Theorem 4.7. *Given an initial configuration with a connected visibility graph, when performing Algorithm 3, the system will converge to a point, in finite time.*

Proof. According to Lemmas 4.4, 4.5, and 4.6, active agents in $C \setminus A$ remain there (or otherwise collapse), while agents in A must necessarily move to $C \setminus A$ as well (or otherwise collapse). Thus, once all agents become active, C will shrink into an area within $C \setminus A$. This will happen in finite time according to the model, and it is true for any of the convex hull's corners. In particular, this is true for the most acute corner. Thus, it can be easily shown that the perimeter of C will decrease by at least a minimum amount $s^* = r\cos\frac{\varphi^*}{2}\left(1-\sin\frac{\varphi^*}{2}\right) > 0$, where

$\varphi^* = \pi\left(1 - \frac{2}{n}\right) < \pi$. Since the initial perimeter is bounded (because the initial configuration is globally connected), it will take a finite number of these finite-time cycles until the perimeter becomes small enough that it will imply that all agents are mutually nearby. At that point, by definition of the algorithm, all agents will collapse into each other once they become active. □

5 Experiments

The formal proofs above guarantee that our algorithms indeed work, yet they do not provide practical bounds on their performance. To gain more insight on their behavior and compare their performance, we performed extensive simulations of the three algorithms. We tried various combinations of values of n, r, and V. We fixed $\sigma = r$ for simplicity as well as fairness, in the sense that the maximum step length became equal in both Algorithms 1 and 2. In all simulations, the initial positions were randomly selected uniformly in a large square area, while ensuring that the visibility graph is connected. During the simulation runs, each agent became active independently with probability 1/2.

The most obvious difference between the three algorithms is in their final or steady state. In Algorithm 1, the agents never stop moving. They contract into a small cluster whose diameter is around the order of r, and keep leaping one over the other ad infinitum. This is a direct result of the agents' inability to measure distances — they have no idea that they all became so close. The cluster is not tied in place and it slowly drifts in the plane. In Algorithm 2, as expected, the agents stop once they are all within an area of diameter r. Finally, in Algorithm 3, the agents collapse into a point.

Another interesting aspect is the behavior of the swarm during the convergence process. In all of our simulations of Algorithm 3, agent collapses were quite rare during the process, with most of them occurring in the very end of the run, once all agents became nearby. This is not surprising given the way we initially positioned the agents. In order for a collapse to occur before the end, there must exist an agent with no far-visible neighbors adjacent to another agent which does see a far agent, which is a somewhat special configuration (Even with the largest setting $r = V/3$, more than 96% of the collapses occurred in the end). As a result, the behavior of Algorithms 2 and 3 was very similar, aside from the slower convergence rate of Algorithm 3 due to its halved step size.

Figure 2 shows a typical run of Algorithm 2. It can be seen that the most significant movement is, as expected, in the outskirts of the swarm, where there are the most agents which are not surrounded by far neighbors (i.e., their wedge angle ψ is less than π). Initially, the swarm assumes a shape with several "tentacles" or "lobes". These tentacles are formed in areas where there was initially a somewhat denser "mass" of agents, which pull more agents from the sparser areas. Thus, we observe what we believe is a reinforcing process of the denser areas, due to the larger number of inactive agents there (in absolute terms). All the while, the agents keep converging in a steady pace until all tentacles contract into an oval body which ultimately contracts into the final configuration. Interestingly,

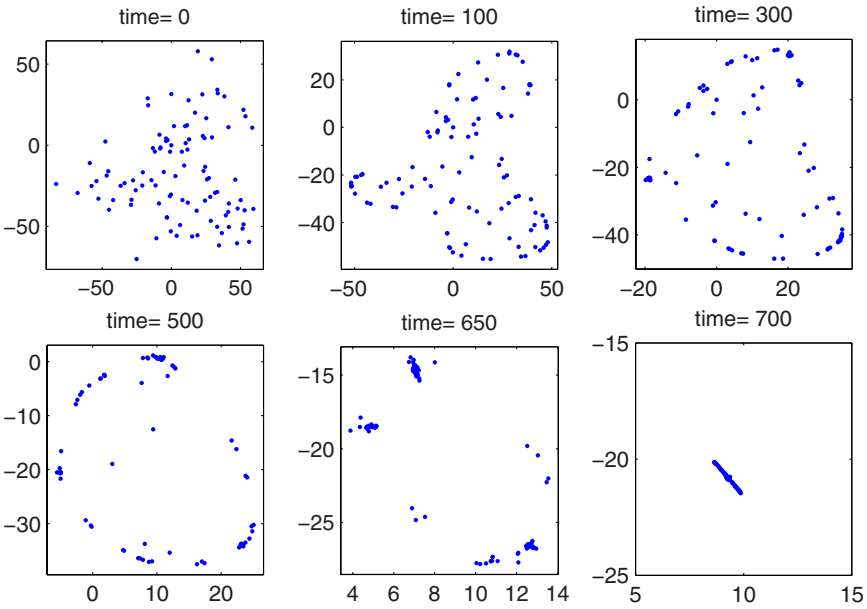
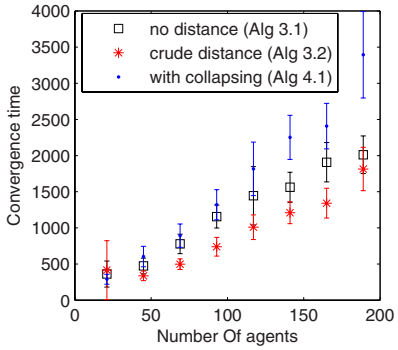
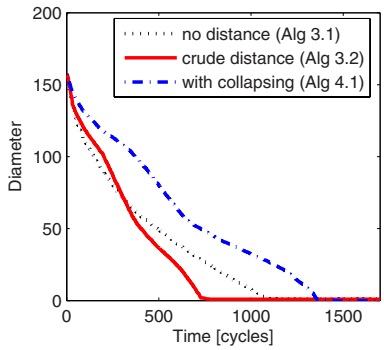


Fig. 2. A typical run of Algorithm 2. Here $n = 100$, $V = 20$ and $r = 1$. Notice that the scale is different between frames.

the swarm typically assumes a different shape with Algorithm 1 (See [16]). Here, the denser areas along the swarm perimeter soon become dense clusters which contract inwards very slowly. Viewed at large scale, the swarm perimeter assumes an almost polygonal form with sharp dense corners and straight sparse edges. The reason that these dense clusters move more slowly is that most of the agents inside are surrounded by their mates and are therefore immobile.



(a) Convergence time statistics for the three algorithms. 12 simulation runs were performed for each value of n.



(b) Diameter vs. time in typical runs of the algorithms. Here $n = 100$, $V = 20$ and $\sigma = r = 1$.

Fig. 3.

Figure 3(a) shows the statistics of our simultations. Algorithm 3 is the slowest, as expected, due to the smaller step size. It is clear that Algorithm 2 provides a significant improvement in convergence time over Algorithm 1, especially for larger swarms. Interestingly, with smaller swarms, there is no clear advantage. Figure 3(b) sheds more light on this issue. It shows the contraction of the swarm's diameter over time in a typical run of each algorithm. Initially, Algorithm 1 is somewhat faster. This is due to the larger step size in Algorithm 1 for most wedge angles ψ (It is $\sigma = r$ versus $r \cos \psi/2$ in Algorithm 2). However, the convergence rate continuously deteriorates whereas the convergence in Algorithm 2 remains more or less constant. This is due to the accumulation of "mass" in the corners of the swarm, which has a more adverse effect on their mobility, as explained above.

6 Conclusion

In this paper we continued to explore the gathering problem under severe distance sensing limitations. Previously, we showed how a swarm of robotic agents can gather without sensing distances at all, but only into a drifting cluster and with suboptimal performance. Now, we have shown that even the crudest form of near/far distance sensing can improve the situation a lot. With the new capability, the swarm was able to contract more swiftly and steadily into a small cluster and stop in place. Finally, introducing the capability of collapsing into nearby agents, the swarm was able to contract into a point.

Future work includes investigation of robustness to noise, failures, and per-agent variations to parameters such as r and V, and development of other algorithms under the crude distance sensing model, such as formation and flocking.

An important part of our work is the contribution to what we feel is an improved model of swarms, through the addition of the strong asynchronicity assumption to the classical semi-synchronous model. Not only does it dismiss the somewhat artificial problem of symmetry breaking (in the context of practical robotics), but it also enables a simple yet powerful approach of proving the termination of an algorithm by showing (by construction) the existence of paths to goal configurations. Strong asynchronicity can be applied to other models, such as the totally-asynchronous model of [19], and possibly to other types of distributed multi-agent systems as well.

References

1. Cieliebak, M., Flocchini, P., Prencipe, G., Santoro, N.: Solving the robots gathering problem. In: Proc. of ICALP 2003 (2003)
2. Gordon, N., Wagner, I.A., Bruckstein, A.M.: Discrete bee dance algorithms for pattern formation on a grid. In: Proc. of IEEE Intl. Conf. on Intelligent Agent Technology (IAT 2003), pp. 545–549 (2003)
3. Schlude, K.: From robotics to facility location: Contraction functions, weber point, convex core. Technical Report 403, CS, ETHZ (2003)

4. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns. SIAM Journal on Computing 28(4), 1347–1363 (1999)
5. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of autonomous mobile robots with limited visibility. In: Ferreira, A., Reichel, H. (eds.) STACS 2001. LNCS, vol. 2010. Springer, Heidelberg (2001)
6. Souissi, S., Défago, X., Yamashita, M.: Gathering asynchronous mobile robots with inaccurate compasses. In: Shvartsman, M.M.A.A. (ed.) OPODIS 2006. LNCS, vol. 4305, pp. 333–349. Springer, Heidelberg (2006)
7. Bruckstein, A.M., Cohen, N., Efrat, A.: Ants, crickets and frogs in cyclic pursuit. Technical Report CIS-9105, Technion – IIT (1991)
8. Bruckstein, A.M., Mallows, C.L., Wagner, I.A.: Probabilistic pursuits on the grid. American Mathematical Monthly 104(4), 323–343 (1997)
9. Marshall, J.A., Broucke, M.E., Francis, B.A.: A pursuit strategy for wheeled-vehicle formations. In: Proc. of CDC 2003, pp. 2555–2560 (2003)
10. Ando, H., Oasa, Y., Suzuki, I., Yamashita, M.: A distributed memoryless point convergence algorithm for mobile robots with limited visibility. IEEE Trans. on Robotics and Automation 15(5), 818–828 (1999)
11. Cohen, R., Peleg, D.: Robot convergence via center-of-gravity algorithms. In: Kralovic, R., Sýkora, O. (eds.) SIROCCO 2004. LNCS, vol. 3104, pp. 79–88. Springer, Heidelberg (2004)
12. Lin, Z., Broucke, M.E., Francis, B.A.: Local control strategies for groups of mobile autonomous agents. IEEE Trans. on Automatic Control 49(4), 622–629 (2004)
13. Melhuish, C.R., Holland, O., Hoddell, S.: Convoying: using chorusing to form travelling groups of minimal agents. Robotics and Autonomous Systems 28, 207–216 (1999)
14. Cortes, J., Martinez, S., Bullo, F.: Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. IEEE Trans. on Automatic Control 51(8), 1289–1298 (2006)
15. Sugihara, K., Suzuki, I.: Distributed algorithms for formation of geometric patterns with many mobile robots. Journal of Robotic Systems 13(3), 127–139 (1996)
16. Gordon, N., Wagner, I.A., Bruckstein, A.M.: Gathering multiple robotic a(ge)nts with limited sensing capabilities. In: Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T. (eds.) ANTS 2004. LNCS, vol. 3172, pp. 142–153. Springer, Heidelberg (2004)
17. Gordon, N., Wagner, I.A., Bruckstein, A.M.: A randomized gathering algorithm for multiple robots with limited sensing capabilities. In: Proc. of MARS 2005 workshop at ICINCO 2005, INSTICC (2005)
18. The Center of Intelligent Systems, Technion IIT web site, `http://www.cs.technion.ac.il/Labs/Isl/index.html`
19. Prencipe, G.: On the feasibility of gathering by autonomous mobile robots. In: Pelc, A., Raynal, M. (eds.) SIROCCO 2005. LNCS, vol. 3499, pp. 246–261. Springer, Heidelberg (2005)
20. Efrima, A., Peleg, D.: Distributed models and algorithms for mobile robot systems. In: van Leeuwen, J., Italiano, G.F., van der Hoek, W., Meinel, C., Sack, H., Plášil, F. (eds.) SOFSEM 2007. LNCS, vol. 4362, pp. 70–87. Springer, Heidelberg (2007)
21. Fredslund, J., Mataric, M.J.: Robot formations using only local sensing and control. In: Proc. of the Intl. Symposium on Computational Intelligence in Robotics and Automation (IEEE CIRA 2001), Banff, Alberta, Canada, pp. 308–313 (2001)