2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

A Distributed Deterministic Spiral Search Algorithm for Swarms

G. Matthew Fricke,[1]* Joshua P. Hecker,[1] Antonio D. Griego,[1] Linh T. Tran,[1] and Melanie E. Moses[1,2,3]

Abstract— **As robot swarms become more viable, efficient solutions to fundamental tasks such as swarm search and collection are required. We propose the distributed deterministic spiral algorithm (DDSA) which generalises a spiral search pattern to robot swarms. While being an effective search strategy in its own right, the DDSA is also a useful point of comparison for other swarm search strategies. Such a benchmark for robot swarm search is currently needed but missing. As a case study, we compare the DDSA to a biologically-inspired central-place foraging algorithm that uses stochastic search, memory, and communication to efficiently collect resources in a variety of different resource distributions.**

I. INTRODUCTION

Many swarm robot applications require the detection and collection of targets by teams of robots. These tasks include planetary surveys [1], land and sea mine clearance [2], pollution mapping by subsurface robots [3], environmental monitoring, survivor location in hazardous environments [4], [5], military applications [6], and agricultural pest control [7]. When the requirement that targets be transported to a single collection point is included in a swarm search problem, it becomes a central place foraging task [8]. Central place foraging tasks include crop harvesting [9] and planetary resource collection [10].

Spiral search patterns for single searchers with stationary targets have been studied extensively and found to have desirable optimality properties [5], [11]–[16]. These properties include detection of the nearest targets first, complete coverage of the area within the spiral, and minimal oversampling. Detection of the nearest targets first is particularly important for central place foraging because it minimises the per-target trip time to the collection point.

Here we present the distributed deterministic spiral algorithm (DDSA). The DDSA generalises a single robot square spiral to any number of robots. The generated spirals are interlocking paths that preserve the determinism of the single robot case and the consequent optimality guarantees. We implement the DDSA using the ARGoS swarm simulator [17] in order to observe how foraging efficiency scales with the number of searchers and targets. The physical robots we simulate are called iAnts [18]. Groups of iAnts are designed to meet the fundamental properties of a swarm described by Brambilia et al [19]. We compare our results to the central place foraging algorithm (CPFA) developed by Hecker and Moses [20], [21].

In both swarm robotics [22] and ecology [23], [24] the performance of systematic search strategies, including spiral search, is assumed to degrade significantly in the presence of error. Stochastic methods which are more resilient to noise are therefore favored [25]. We investigate the impact of positional error on the performance of the DDSA and compare our results to the CPFA.

Central place foraging algorithms do not have a baseline of comparison. This makes it difficult to evaluate the effectiveness of foraging algorithms. Swarm algorithms tend to be probabilistic and contingent on the hardware or simulation in which they are implemented. This makes improvements in performance difficult to compare across systems and hard to describe analytically. We propose the use of the DDSA as a point of comparison for other central place foraging algorithms. The DDSA has two essential properties that make it a good candidate as a baseline algorithm: 1) it is simple from a theoretical point of view, being deterministic and having behaviour definable using a simple recurrence relation, and 2) in the error free case, it guarantees collection of the nearest targets first, complete coverage, and minimal repeated sampling.

II. RELATED WORK

In addition to the single searcher work listed in the introduction, Ryan and Hedrick describe a square search pattern carried out with a single helicopter [26]. This search pattern is defined in Appendix H of the Coast Guard Operating Manual and is similar to the DDSA single searcher base case.

Baeza-Yates and Schott describe a multi-agent spiral search algorithm in which agents begin at a central point [27]. However, because they use a circular spiral the searchers diverge from one another over time. As a result, the approach is only able to reliably detect lines rather than point targets placed at arbitrary locations in the plane.

Parallel spiral search approaches have also been implemented in which each searcher performs an independent single agent spiral spatially removed from the other members of the swarm [28], [29], a behaviour observed in ants in our own lab.[†] Stocastic spiral search patterns have also been observed as a central place foraging stratgey of desert ants (*Cataglyphis fortis*) [30]. These ants inhabit salt pans, which are flat and obstacle free compared to most natural landscapes.

[1]Computer Science Department, [2]Biology Department, University of New Mexico, Albuquerque, USA.[3]External Faculty Member, Santa Fe Institute, Santa Fe, USA.
*Correspondence: `mfricke@cs.unm.edu`

[†]Spiralling Ant Video: `youtu.be/N46u0xLl56o`

The Multiple Robots Internal Spiral Coverage algorithm is a solution that guarantees complete coverage of a environment by partitioning the space equally among multiple robots [31]. This approach differs from the DDSA because the robot search paths are discretized by a grid structure and robots are assigned to regions within the grid.

In work closely related to our own, Skubch decribes a "proof of concept" approach for generating a circular distributed spiral for multiple robots [32]. A dynamic constraint optimisation function uses stateful-feedback and a shared datastructure to coordinate the movement of robots in the swarm. Intriguingly, this dynamic constraint update at each time step allows the redistribution of robots in the event of robot failure, but also results in robots randomly switching between each other's spiral paths. In our own approach the search pattern is predetermined and robots do not comminicate with one another during search or maintain a shared datastructure.

López and Maftuleac describe a deterministic search strategy for idealized searchers that in the case of 2 and 4 searchers results in an interlocking spiral [33]. When the number of searchers exceeds 4, the algorithm partitions the space into expanding wedges. This strategy requires searchers to perform a right angle turn every step. They find this approach to be robust to error in target detection and searchers with differing speeds.

The CPFA with which we compare the DDSA is an ant-inspired algorithm [20], [21]. As robots search the environment they probabilistically place waypoints at locations with high target density. These waypoints influence other members of the swarm towards searching areas where more targets are expected to be found. When not using waypoints robots perform a random walk with decaying correlation. This strategy ensures that areas near where a target was previously found are searched intensively but that if nothing is found the robot quickly moves to a new area. The parameters that govern the CFPA, such as the probability of placing waypoints as a function of local target density, are optimised with a genetic algorithm.

III. METHODS

A. The Algorithm

The DDSA specifies the interlocking spirals for a group of robots by calculating how far each robot must travel along each edge of their particular spiral (Figure 1). Calculating the spiral paths requires knowing 1) how many robots there are so enough room is left for all of them, 2) the target detection range of the robots so the gap between spirals is eliminated, 3) how far into the spiral the robot is, since the spiral expands over time, and 4) the index of each robot in a predefined order.

Let D_H be a function that determines how far the current robot should travel in a given cardinal direction, H, for a particular circuit count, c. The circuit count is the number of times a robot has completed movement in all four directions in N, E, S, W order. Robots move away from the central location and order themselves on the 0th circuit. Formally,

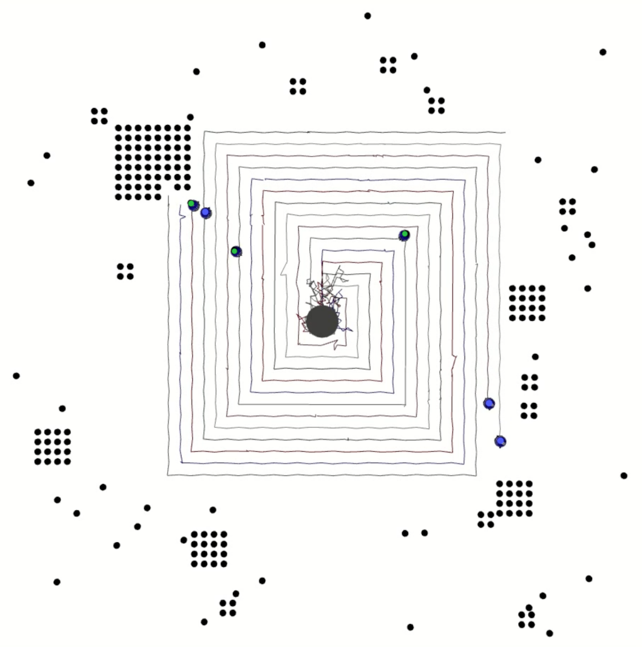


Fig. 1: The DDSA running in ARGoS, overhead view. The robots search a continuous plane employing a spiral search pattern beginning at a central collection point. Targets are shown as black dots arranged in a partially clustered distribution. Robots are marked with a blue or green dot. Robots with a green dot are carrying targets, those with a blue dot are searching. Coloured lines are the paths of the various robots. Paths taken to and from the central collection point are not displayed. The simulation is perfomed in continuous space and robot paths and target placements are not lattice bound.

$D_H(i, c, R) : I \times \mathbb{N}^0 \times \mathbb{N}^0$ where $I = \{i \mid 1 \leq i \leq R\}$ is the index of the current robot, c is the current circuit, and R is the total number of robots. The north (N) and east (E), and south (S) and west (W) movement cases are symmetric, so $D_{\mathrm{N}} = D_{\mathrm{E}}$ and $D_{\mathrm{S}} = D_{\mathrm{W}}$.

$$D_{\mathrm{N}}(i, c, R) = \begin{cases} i & c = 0 \\ D_{\mathrm{N}}(i, 0, R) + R + i & c = 1 \\ D_{\mathrm{N}}(i, c-1, R) + 2R & c > 1 \end{cases} \tag{1a}$$

$$D_{\mathrm{S}}(i, c, R) = \begin{cases} D_{\mathrm{N}}(i, c, R) + i & c = 0 \\ D_{\mathrm{N}}(i, c, R) + R & c > 0 \end{cases} \tag{1b}$$

Requiring each robot to know the swarm size and its index implies global knowledge which would theoretically violate a principle of swarm design. Fortunately there are a number of consensus addressing algorithms for multi-robot systems that allow the swarm to determine its size and introduce an ordering [34].

The gap between adjacent search paths, g, must be narrow enough so that the target detection ranges of robots on adjacent paths overlap to guarantee complete coverage; however, overlap should be minimized to avoid resampling of the same location. The target detection range, r, is $13\,\mathrm{cm}$ in our robots, suggesting a gap of $26\,\mathrm{cm}$. However, at the corners of the square spirals the distance between paths increases to $\sqrt{2g^2}$. Therefore, in order to guarantee complete collection within the spiral, we set $g = \sqrt{(2r)^2/2} \approx 18\,\mathrm{cm}$ to compensate.

Let S be the set of searcher positions, along with D_H and g we can define interlocking square spirals for each robot and the state machine given in Algorithm 1.

Algorithm 1 DDSA

```
    ▷ Distributed across robots
 1: for all robots $i \leftarrow 1$ to $R$ do
        ▷ Create a spiral pattern to follow and store it
 2:     for $c \leftarrow 0$ to NCircuits do
 3:         Q.enqueue($\langle 0, gD_N(i,c,R)\rangle$)
 4:         Q.enqueue($\langle gD_E(i,c,R), 0\rangle$)
 5:         Q.enqueue($\langle 0, -gD_S(i,c,R)\rangle$)
 6:         Q.enqueue($\langle -gD_W(i,c,R), 0\rangle$)
 7:     end for
        ▷ Start at collection point and perform spiral
 8:     while $\neg$Q.empty() do
 9:         $w \leftarrow s+$Q.dequeue()
10:         Move toward $w$
11:         if target found at current location $s$ then
12:             Return to collection point with target
13:             if at collection point then
14:                 Deposit target
15:                 Return to location $s$
16:             end if
17:         end if
18:     end while
19: end for
```

B. Robot Simulation

We implement the DDSA and CPFA using the ARGoS swarm robot simulator [17].[‡] ARGoS supports high fidelity ODE physics engines which allow the accurate detection of robot collisions. We use the 2D physics solver provided with ARGoS running at 320 updates per second for our simulations.

The parameters for the robots are informed by those of the physical iAnt robots designed and built in our lab [20]. Basing the simulated robots on physical robots allows us to choose positional error that closely matches our experience with real hardware. In order to represent iAnt hardware, the simulated robots are $8\,\mathrm{cm}$ in radius and have a downward facing camera capable of seeing directly below the robot. Targets have a radius of 5 cm, together with the $8\,\mathrm{cm}$ robot viewing area this gives a value of $13\,\mathrm{cm}$ for parameter r in the DDSA. Robots have a forward movement rate of $16\,\mathrm{cm\,s^{-1}}$ and a rotation rate of $8\,\mathrm{cm\,s^{-1}}$ or approximately $1\,\mathrm{rad\,s^{-1}}$. Robots move $8\,\mathrm{cm}$ towards their goal locations between reorientations.

[‡]The software used in this work is available on GitHub:
`github.com/BCLab-UNM/DDSA-ARGoS/release/0.2-beta`
`github.com/BCLab-UNM/CPFA-ARGoS/release/0.1-beta`

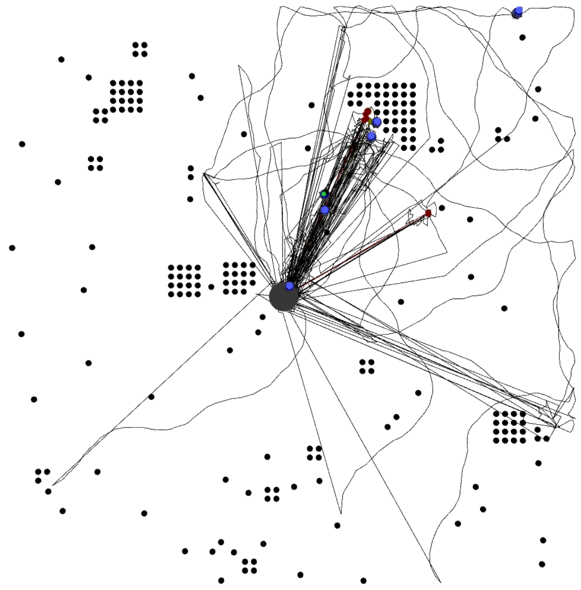


Fig. 2: The CPFA running in ARGoS, overhead view. The partially clustered distribution of targets is shown as black dots, robots green or blue dots, lines indicate the paths taken by searchers during the experiment. Red dots indicate waypoints used by searchers to communicate the location of dense target clusters to the swarm.

We simulate error by applying Gaussian noise to robot destinations. To replicate our observations that the iAnt robots accumulate error linearly with distance [18], we make the standard deviation increase with the distance from the robot's current position to its destination position, x. We multiply the standard deviation by a noise coefficient, e, in order to change noise severity. Therefore, our noise variates, v, are generated by:

$$v \sim \mathcal{N}(0, \sigma^2) \text{ where } \sigma = d(s,x) \times e \tag{2}$$

We do not apply positional noise to robots returning to the central location point for the DDSA or CPFA. We assume that the collection point is marked by a beacon following our previous studies with iAnt hardware [20].

C. Experimental Setup

The problem domain for central place foraging algorithms is target placements within a plane. Each problem instance consists of a set of coordinates representing the locations of targets. To be useful a search algorithm must work effectively across a wide variety of potential target configurations. We measure the performance of the DDSA and CPFA by measuring target collection times over many randomly chosen problem instances. While the DDSA is a deterministic algorithm, its performance on any particular problem instance varies according to the particular placement of targets. Collisions between robots can also introduce non-deterministic effects into the search process.

In our experiments, we place 256 targets in a $100\,\mathrm{m}^2$ arena according to three random distributions 1) uniform, 2) partially clustered, and 3) clustered. The uniform distribution places targets at all locations in the search arena with equal probability. The partially clustered distribution follows a power law with 1 cluster of 16, 4 clusters of size 64, 16 of size 4, and 64 single targets. The clustered distribution consists of 4 target clusters with 64 targets per cluster. If a cluster or target location is occupied a new uniform random location is chosen. We choose the partially clustered distribution of targets as our default target distribution because naturally occurring targets tend to occur in a variety of cluster sizes [35].

All experiments last $30\,\mathrm{min}$, except experiments measuring complete collection time, which run indefinitely. In all figures 25 experimental runs contribute to each data point, except in Figure 3 where we use 50 replicates.

IV. RESULTS

A. Performance

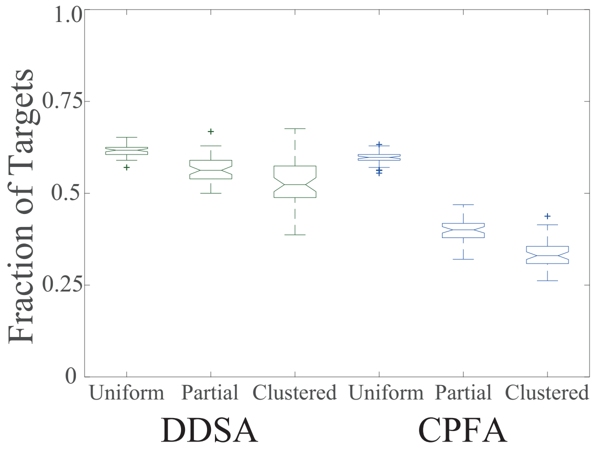


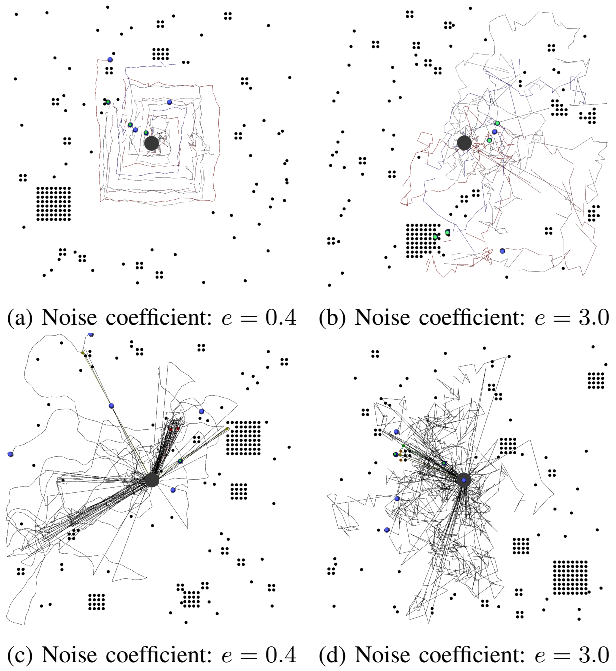
Fig. 3: Comparison of DDSA and CPFA performance for 3 target distributions. Experiments are $30\,\mathrm{min}$ in a $100\,\mathrm{m}^2$ search arena with 6 robots and without noise.

Our experiments show that the DDSA performs at least as well as the CPFA in the 6 robot, no error case (Figure 3). This confirms our expectation that the DDSA has desirable search and collection properties. Beyond those already discussed (closest targets first, complete collection and minimal oversampling), the DDSA always returns to the location at which it last found a target. This is so that the search spiral can be rejoined at the point it was interrupted, but it has the useful side effect of sending robots back to clusters of targets. The process of returning to the location of the last-found target is called site fidelity, and is a common search strategy in ants [36], [37]. This strategy is also used by the CPFA.

The DDSA partitions the search space among searchers equally after the 0th circuit. This is reflected in the order of performance we observed in Figure 3. The uniform target distribution results in an equal allocation of robots to the target collection task, partially clustered less so, and the clustered case least of all. The unequal allocation of robots to targets results in a decrease in performance. Additionally, when a cluster of targets is encountered collisions between robots increase near the cluster.

Similarly, in the CPFA experiments uniform targets are collected fastest, followed by partially clustered targets, and clustered targets are collected slowest. This pattern is reversed from Hecker and Moses [20], likely because that prior work did not consider collisions. Once a cluster is detected the CPFA can take advantage of it by recruiting robots to that location, but the initial time to discover a cluster, and the increase in collisions at clusters, offsets this advantage.

B. Robustness



(a) Noise coefficient: $e = 0.4$ (b) Noise coefficient: $e = 3.0$

(c) Noise coefficient: $e = 0.4$ (d) Noise coefficient: $e = 3.0$

Fig. 4: Effect of positional noise on the DDSA and CPFA search patterns. The DDSA is shown in panels (a) and (b) and the CPFA in panels (c) and (d). Black dots are targets, green and blue dots indicate the current location of robots and lines are the paths taken by robots.

Degradation of the search pattern under positional noise results in only modest decreases in performance. The number of collected targets is reduced by only 15% between the error free case and our maximum error case (Figure 5).

In the maximum error case the positional noise is substantial. For example, a robot travelling $10\,\mathrm{cm}$ has destination positional error with standard deviation $30\,\mathrm{cm}$ (Figure 4b). This robustness to error is due to robots progressively searching locations close to the collection point even with positional noise. Additionally, positional error in one robot may be compensated for by noise in adjacent robots. When the swarm is large, tags that are missed by one robot that is out of alignment with its spiral are likely to be picked up by robots noisily following adjacent paths (Figure 4a).

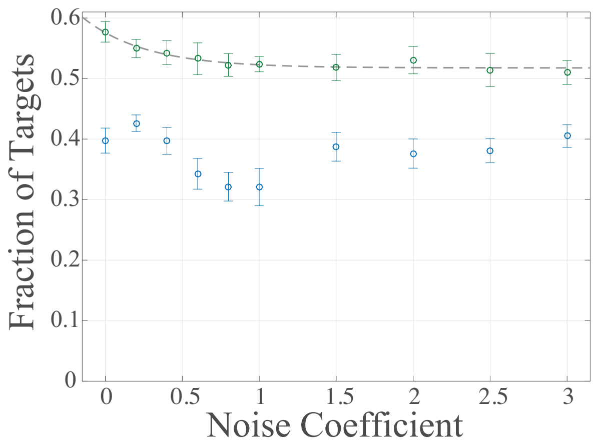


Fig. 5: In green, DDSA performance scaling with increasing error. Experiment time is 30 min. Dashed line is an exponential decay fit with $R^2 = 0.926$. In blue, CPFA performance scaling with increasing error. The best linear fit has $R^2 = 0.004$ indicating that the noise coefficient, e, explains very little of the variance in performance. Circles are means and bars are the 95% confidence intervals. We use a partially clustered distribution of targets in a $100\,\mathrm{m}^2$ arena.

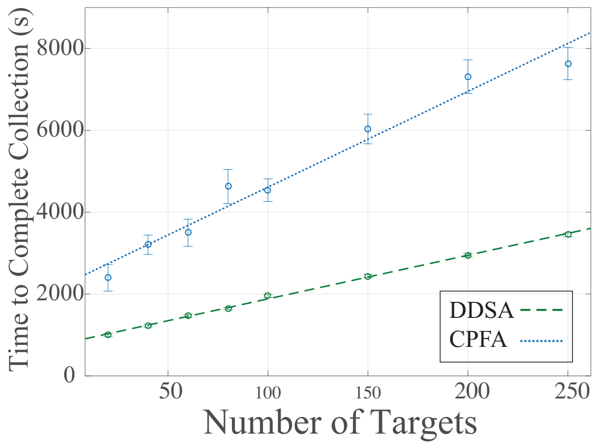


Fig. 6: In green, with dashed fit line, time for the DDSA to collect all targets vs. number of targets. The dashed line is the linear fit with slope 10.67 seconds per target, $R^2 = 0.998$. In blue, with dotted fit line, time for the CPFA to collect all targets vs. number of targets. Dotted line is a linear fit, slope 23.4 seconds per target, $R^2 = 0.968$. The circles are means and bars are the 95% confidence intervals. We use a uniform distribution of targets in a $100\,\mathrm{m}^2$ arena with 6 robots.

The CPFA is also robust to error. Compared to the CPFA without positional noise shown in Figure 2, search paths with $e = 0.4$ are qualitatively unchanged in Figure 4c. In the maximum error case, $e = 3.0$, shown in Figure 4d an increase in path tortuosity is apparent. Noise in the CPFA does not systematically decrease performance (Figure 5).

C. Complete Collection

The DDSA guarantees complete collection of targets within the search spiral in the noise free case and as a result performs complete collection tasks faster than the CPFA (mean decrease in collection time is 59.2%). In the CPFA the time to find uniform targets increases exponentially as the number of remaining targets decreases [21]. However, the time to complete collection scales linearly with the number of targets. For each additional target the time for DDSA collection increases by 10.67 s compared to 23.4 s per additional target with the CPFA (Figure 6).

The 95% confidence interval is tight relative to the stochastic CPFA, as expected for a deterministic algorithm.

D. Scaling with the Number of Robots

The DDSA outperforms the CPFA for swarms consisting of between 1 and 15 robots. For swarms with between 20 and 30 robots DDSA performance drops below that of the CPFA (Figure 7). The DDSA performance curve follows a parabola reaching its maximum between 15 and 20 searchers. Degradation of performance is due to crowding at the collection point (Figure 8). That crowding at the collection point is the main driver for degradation in performance is supported by our observation of linear scaleup when robots are not required to return to the collection point (data not shown) and in previous work [38].

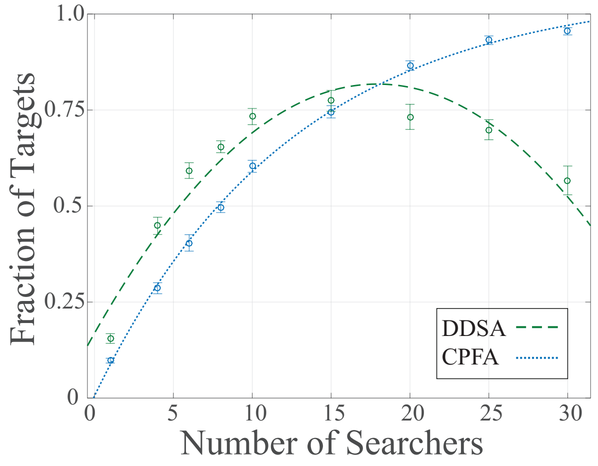


Fig. 7: In green, with dashed fit line, DDSA performance scaling with the number of searchers. The dashed line is a parabolic fit with $R^2 = 0.998$. In blue, with dotted fit line, CPFA performance scaling. The dotted line is a negative exponential fit with slope with $R^2 = 0.922$. Circles are means and bars are the 95% confidence intervals. We use the partially clustered distribution of targets in a $100\,\mathrm{m}^2$ arena with a 30 min time limit.

In the CPFA we observe a negative exponential increase in performance (Figure 7). It is possible that the CPFA is also following a parabolic curve with an inflection point at a much higher number of robots than in the DDSA. Lower levels of congestion at the collection point are likely due to the stochastic nature of the CPFA which reduces the likelihood of robots contending for the same location at the same point in time.

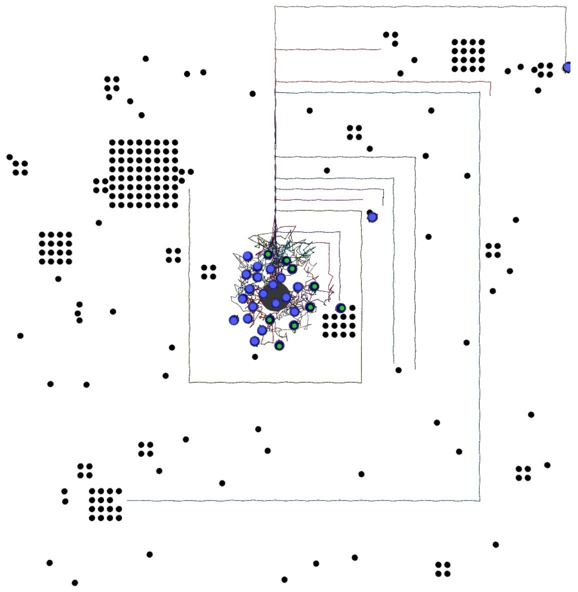


Fig. 8: Crowding in the DDSA degrades performance. State after $30\,\mathrm{min}$ with 30 robots.

E. Worst Case Performance

Since the DDSA always collects items close to the collection point first an adversary that placed all targets at the edge of the search arena would force the DDSA to maximise its search and collection time. For our examples with a $100\,\mathrm{m}^2$ arena, 6 searchers, 256 targets, a gap of $18\,\mathrm{cm}$ and a robot moving at $16\,\mathrm{cm\,s}^{-1}$ it takes approximately $4\,\mathrm{h}$ to collect all the targets, compared to the observed mean time of approximately $1\,\mathrm{h}$. Since the DDSA is a preplanned deterministic algorithm, an adversary could force the DDSA to maximise its collection time by placing all targets along the spiral path of a single searcher. The CPFA, in contrast, is a stochastic algorithm that adapts to the distribution of targets. As such, design of a worst case scenario for the CPFA is much more difficult.

F. Comparison to a Perfect Algorithm

We can calculate the performance of a hypothetical perfect algorithm in which the location of all targets are known to the swarm *a priori*. The expected distance from the collection point $p = (0, 0)$, to a uniformly distributed target, within the $100\,\mathrm{m}^2$ square arena is, without loss of generality, the expected distance to a target, $t = (x, y)$, in the arena's positive quadrant. Numerically solving the double integral over the probability of each Euclidean distance, $\frac{1}{5^2}\int_0^5\int_0^5 d(p,t)dxdy$, gives an expected distance of $3.826\,\mathrm{m}$ to each target. For 250 targets the perfect algorithm, neglecting collisions, will collect all targets using a single robot in expected time $\mathbf{E}\left[C_{\text{time}}\right] = \frac{250\times2\times3.826\,\mathrm{m}}{0.16\,\mathrm{m\,s}^{-1}} + 250\left(\pi + \frac{\pi}{2}\right)\mathrm{s} = 13\,241\,\mathrm{s}$, where the first term is the linear travel time and the second is the turning time. Therefore, in the 6 robot case complete collection will take an expected $2207\,\mathrm{s}$. The mean time taken by the DDSA to collect 250 targets with 6 robots is 3447, 95% CI [3399, 3493] s, an increase of approximately 56.2% over the theoretical minimum.

The mean distance from the centre of a square of width 10 to its perimeter is 5.74 [39]. Substituting this value for 3.826 in the expected time formula above yields $19\,116\,\mathrm{s}$ in the single robot case. So for 6 robots we have a collection time of approximately $53\,\mathrm{min}$. This gives a worst case increase in DDSA collection time over the perfect algorithm of 353%.

V. CONCLUSIONS

We show that desirable properties of the single agent square spiral, extensively demonstrated in previous work, can be extended to multiple robots. The DDSA has optimality properties which make it ideal for use as a central place foraging benchmark: guaranteed collection of nearest objects first, complete collection, and minimal oversampling. Benchmark algorithms should provide an efficient and theoretically tractable point of comparison for more complex algorithms. By comparing the CPFA to the DDSA we have a better understanding of the CPFA's strengths and weaknesses.

Adaptive search patterns such as the CPFA take advantage of information about target clusters. Doing so increases target detection rates but does not minimise trip time. This is highlighted by the relatively good performance of the DDSA (Figure 3). This suggests that a modification to the CPFA, the use of distance information when deciding whether a robot should use a waypoint, could be beneficial.

While the DDSA is surprisingly resilient to error the CPFA is even less affected (Figures 4c and 5). This suggests that the DDSA can be an effective strategy even for robots with limited ability to localise.

The DDSA solves the complete collection problem optimally, in that there is no redundancy in the search pattern and it guarantees collection of all targets within the spiral. In the 6 robot case, the DDSA mean complete collection time is only 56.2% greater than the theoretical perfect algorithm, which has perfect prior knowledge of all target locations and no collisions. In comparison the CPFA's stochastic strategy takes much longer to collect all targets; using the DDSA as a benchmark provides a point of comparison that allows us to quantify this difference (Figure 6).

Two factors make finding scalable solutions to central place foraging difficult. Congestion at the central collection point and the mean distance to the targets both grow with the rate of target collection, which in turn grows with the number of robots. However, the stochastic nature of the CPFA means that it does not suffer as much as the DDSA from the central-point collision bottleneck. This results in the CPFA outperforming the DDSA when the swarm size exceeds 20 robots (Figure 7). The congestion in the beginning setup phase of the DDSA (Figure 8) could be mitigated by staggering a time delay when robots begin the spiral, or by moving directly to their positions in circuit 0 without travelling to the centre of the map first. Generalisation to multiple collection points allows for more scalable solutions such as the multiple-place foraging algorithm (MPFA) [40].

The DDSA provides both theoretical and practical advantages as a general search algorithm, central place foraging strategy, and a benchmark. We expect this approach will find applications in a wide variety of robot swarm tasks.

ACKNOWLEDGMENTS

We gratefully acknowledge funding from a James S. McDonnell Foundation Complex Systems Scholar Award and NASA MUREP #NNX15AM14A for the UNM NASA Swarmathon. Thanks to Elizabeth Esterly for her video of an ant performing a spiral. Reviewer comments and corrections were invaluable.

REFERENCES

[1] W. Fink, J. M. Dohm, M. A. Tarbell, T. M. Hare, and V. R. Baker, "Next-generation robotic planetary reconnaissance missions: a paradigm shift," *Planetary and Space Science*, vol. 53, no. 14, pp. 1419–1426, 2005.

[2] T. R. Weber, "An Analysis of Lemmings: A Swarming Approach to Mine Countermeasures in the VSW/SZ/BZ.," tech. rep., DTIC Document, 1995.

[3] H. Hu, J. Oyekan, and D. Gu, "A school of robotic fish for pollution detection in port," *Biologically Inspired Robotics (Y. Liu and D. Sun, eds.)*, pp. 85–104, 2011.

[4] A. Birk and S. Carpin, "Rescue robotics - a crucial milestone on the road to autonomous systems," *Advanced Robotics*, vol. 20, no. 5, pp. 595–605, 2006.

[5] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a cameraequipped mini UAV," *Journal of Field Robotics*, vol. 25, no. 12, pp. 89–110, 2008.

[6] J. Love, W. Amai, T. Blada, C. Little, J. Neely, and S. Buerger, "The Sandia architecture for heterogeneous unmanned system control (SAHUC)," in *Proc. SPIE 9464, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR VI*, International Society for Optics and Photonics, 2015.

[7] K. Tamura and K. Naruse, "Unsmooth field sweeping by Balistic random walk of multiple robots in unsmooth terrain," in *Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on*, pp. 585–589, IEEE, 2014.

[8] A. F. T. Winfield, "Foraging robots," in *Encyclopedia of complexity and systems science*, pp. 3682–3700, Springer, 2009.

[9] C. W. Bac, E. J. Henten, J. Hemming, and Y. Edan, "Harvesting Robots for Highvalue Crops: Stateoftheart Review and Challenges Ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014.

[10] S. Ramsey, "NASA Awards Grant to Manage Swarmathon Challenge." 2015.

[11] J. L. Bentley, B. W. Weide, and A. C. Yao, "Optimal expected-time algorithms for closest point problems," *ACM Transactions on Mathematical Software (TOMS)*, vol. 6, no. 4, pp. 563–580, 1980.

[12] S. Burlington and G. Dudek, "Spiral search as an efficient mobile robotic search technique," in *Proceedings of the 16th National Conf. on AI, Orlando Fl*, 1999.

[13] R. A. Baeza-yates, J. C. Culberson, and G. J. E. Rawlins, "Searching in the plane," *Information and computation*, vol. 106, no. 2, pp. 234–252, 1993.

[14] E. Langetepe, "On the optimality of spiral search," in *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pp. 1–12, Society for Industrial and Applied Mathematics, 2010.

[15] M. A. A. ElHadidy, "Optimal spiral search plan for a randomly located target in the plane," *International Journal of Operational Research*, vol. 22, no. 4, pp. 454–465, 2015.

[16] H. M. A. Gabal and M. A. A. El-Hadidy, "Optimal searching for a randomly located target in a bounded known region," *International Journal of Computing Science and Mathematics*, vol. 6, no. 4, pp. 392–403, 2015.

[17] C. Pinciroli, V. Trianni, R. OGrady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, and F. Ducatelle, "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems," *Swarm intelligence*, vol. 6, no. 4, pp. 271–295, 2012.

[18] J. P. Hecker, K. Stolleis, B. Swenson, K. Letendre, and M. E. Moses. Evolving Error Tolerance in Biologically-Inspired iAnt Robots. In *Proceedings of the Twelfth European Conference on the Synthesis and Simulation of Living Systems (Advances in Artificial Life, ECAL 2013)*, pages 1025–1032, 2013.

[19] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[20] J. P. Hecker and M. E. Moses, "Beyond pheromones: evolving error-tolerant, flexible, and scalable ant-inspired robot swarms," *Swarm Intelligence*, vol. 9, no. 1, pp. 43–70, 2015.

[21] J. P. Hecker, J. C. Carmichael, and M. E. Moses, "Exploiting clusters for complete resource collection in biologically-inspired robot swarms," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[22] M. Keeter, D. Moore, R. Muller, E. Nieters, J. Flenner, S. E. Martonosi, A. L. Bertozzi, A. G. Percus, and R. Levy, "Cooperative search with autonomous vehicles in a 3d aquatic testbed," in *American Control Conference (ACC), 2012*, pp. 3154–3160, IEEE, 2012.

[23] A. M. Reynolds, A. D. Smith, R. Menzel, U. Greggers, D. R. Reynolds, and J. R. Riley, "Displaced honey bees perform optimal scale-free search flights," *Ecology*, vol. 88, no. 8, pp. 1955–1961, 2007.

[24] F. Papi, *Animal homing*. Springer Science & Business Media, 2012.

[25] Y. B. Sebbane, *Lighter than air robots: guidance and control of autonomous airships*, vol. 58. Springer Science & Business Media, 2011.

[26] A. Ryan and J. K. Hedrick, "A mode-switching path planner for UAV-assisted search and rescue," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pp. 1471–1476, IEEE, 2005.

[27] R. Baeza-Yates and R. Schott, "Parallel searching in the plane," *Computational Geometry*, vol. 5, no. 3, pp. 143–154, 1995.

[28] A. T. Hayes, A. Martinoli, and R. M. Goodman, "Swarm robotic odor localization," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 2, pp. 1073–1078, IEEE, 2001.

[29] O. Feinerman, A. Korman, Z. Lotker, and J.-S. Sereni, "Collaborative search on the plane without communication," in *Proceedings of the 2012 ACM symposium on Principles of distributed computing*, pp. 77–86, ACM, 2012.

[30] M. Müller and R. Wehner, "The hidden spiral: systematic search and path integration in desert ants, Cataglyphis fortis," *Journal of Comparative Physiology A*, vol. 175, no. 5, pp. 525–530, 1994.

[31] Z. B. Hao, N. Sang, and H. Lei, "Cooperative Coverage by Multiple Robots with Contact Sensors," in *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, pp. 543–548, IEEE, 2008.n

[32] H. Skubch, *Modelling and Controlling of Behaviour for Autonomous Mobile Robots*. Springer Science & Business Media, pp. 220–224,2012.

[33] A. López-Ortiz and D. Maftuleac. Optimal Distributed Searching in the Plane with and without Uncertainty. In *International Workshop on Algorithms and Computation*, pages 68–79. Springer, 2016.

[34] M. R. Thoppian and R. Prakash. A distributed protocol for dynamic address assignment in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(1):4–19, 2006.

[35] J. B. Dunning, B. J. Danielson, and H. R. Pulliam, "Ecological processes that affect populations in complex landscapes," *Oikos*, pp. 169–175, 1992.

[36] B. D. Beverly, H. McLendon, S. Nacu, S. Holmes, and D. M. Gordon, "How site fidelity leads to individual differences in the foraging activity of harvester ants," *Behavioral Ecology*, vol. 20, no. 3, pp. 633–638, 2009.

[37] T. P. Flanagan, K. Letendre, W. R. Burnside, G. M. Fricke, and M. E. Moses. Quantifying the effect of colony size and food distribution on harvester ant foraging. *PLoS ONE*, 7(7), 2012.

[38] G. M. Fricke, J. P. Hecker, J. L. Cannon, and M. E. Moses. Immune-inspired search strategies for robot swarms. *Robotica*, 34(08):1791–1810, 2016.

[39] W. W. Johnson, *A treatise on the integral calculus founded on the method of rates*, Wiley & sons, New York, pp. 224, 1907.

[40] Q. Lu, J. P. Hecker, and M. E. Moses. The MPFA: A Multiple-Place Foraging Algorithm for Biologically-Inspired Robot Swarms. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016 (in press).