2017 14th Conference on Computer and Robot Vision

# LOST Highway: a Multiple-Lane Ant-Trail Algorithm to Reduce Congestion in Large-Population Multi-Robot Systems

Alaa Eldin Abdelaal
*School of Computing Science*
*Simon Fraser University*
*Burnaby, BC, Canada*
*alaa_eldin_abdelaal@sfu.ca*

Maram Sakr
*School of Engineering Science*
*Simon Fraser University*
*Burnaby, BC, Canada*
*msakr@sfu.ca*

Richard Vaughan
*School of Computing Science*
*Simon Fraser University*
*Burnaby, BC, Canada*
*vaughan@sfu.ca*

***Abstract*—We propose a modification of a well-known ant-inspired trail-following algorithm to reduce congestion in multi-robot systems. Our method results in robots moving in multiple lanes towards their goal location. Our algorithm is inspired by the idea of building multiple-lane highways to mitigate traffic congestion in traffic engineering. We consider the resource transportation task where autonomous robots repeatedly transport goods between a food source and a nest in an initially unknown environment. To evaluate our algorithm, we perform simulation experiments in several environments with and without obstacles. Compared with the baseline SO-LOST algorithm, we find that our modified method increases the system throughput by up to 3.9 times by supporting a larger productive robot population.**

## I. Introduction

In swarm robotics, a large number of simple robots is used to perform a task instead of relying only on a single complex robot. This has a lot of potential in many applications. For example, a multi-robot system can be utilized in the resource transportation task where simple robots repeatedly transport resources between two or more locations in a warehouse. Previous work proposes different approaches for this task in the context of mutli-robot systems. In [1], the authors propose a ant-trail algorithm called LOST, where robots lay logical waypoints, called crumbs, towards their goal locations and broadcast them. These waypoints are virtual pheromones that imitate the way real ants behave in their environments. The waypoints result in trails between a food source and a home location. These trails are optimized online, leading consequently to a near-optimal single trail. The LOST method is designed to be practical for real robots under certain constraints. In [2], the authors show that with large population of robots, this single trail becomes congested and physical interferences between robots can damage the system performance. As a result, the authors propose the Spread-Out LOST (SO-LOST) algorithm which results in having two spatially-separated trails; one goes from the home location to the source and the other goes in the opposite direction. Moreover, in [3] the authors study the effect of changing the field of view (FOV) of the robots in the original LOST algorithm on the system performance, They conclude that by limiting the FOV, robots tend to construct different trails in the environment which increases the overall system performance. In this paper, we present a method to establish multiple lanes/trails in order to better distribute the robots in the environment, reduce the density of the trails and the amount of inter-robot interference, and thereby increase the system throughput in large populations. The contribution is to show how SO-LOST and related algorithms can be scaled up to larger, more dense populations than previously possible.

## II. Related Work

Several previous authors have investigated the use of spatial patterns of pheromones in multi-robot systems. Some of them proposed using physical forms of pheromones while others made use of logical ones. In [4], Khaliq and Saffiotti used RFID tags to guide a team of ePuck robots through their navigation in an apartment. Similar use of RFID tags as pheromones was proposed in [5] and [6]. In addition, Mayet *et al.* [7] used phosphorescent to mimic the foraging behavior of the ants. Furthermore, some earlier work like [8] and [9], used chemical substances as pheromones to mark the world. The problem with all these approaches is that they are difficult to use in practical robot systems, despite their incredible success in real ants. A more promising approach is to implement purely informational *virtual pheromones*. For example, Hoff *et al.* [10] utilized bidirectional communication between robots to exchange relative position information and hence implement virtual pheromones. They proposed two algorithms for the foraging task. The first was a gradient-based algorithm which is suitable when the food source is near the initial locations of the robots. The second one was a sweeper algorithm at which robots sweep the environment searching for far away food sources. They also presented a switching algorithm between these two algorithms along with a random walk behavior according to the environment. The problem with all these algorithms is that some of the robots are supposed to remain stationary at some point and they act as landmarks or waypoints for other robots (which they call walkers). This prevents the

978-1-5386-2818-8/17 $31.00 © 2017 IEEE
DOI 10.1109/CRV.2017.24

system from achieving its maximum possible throughput. The idea of using bidirectional communication to implement virtual pheromones was also presented earlier in [11] and more recently in [12].

The effect of spatial interference between robots on the overall system performance in multi-robot systems was studied in [13]. This paper showed that while the system's overall performance increases by adding more robots, the individual robot's performance decreases. One way to solve the spatial interference problem is by enforcing physical separation between robots. In the context of the resource transportation task, some work proposes using task partitioning between robots. This is done by assigning a robot or some robots to a specific area in the working environment. The role of the robots is to pick the resources from and drop them into the working area of another robot. The assignment of robots to different working areas can be done in a static or dynamic fashion [14], [15], [16]. The problem with these approaches is that the trajectory length to perform the task increases with the numbers of robots. This consequently prevents the system from reaching its optimal performance. Scheidler *et al.* [17] proposed congestion control strategies to resolve conflicts between robots. However, the proposed methods require modifications either in the environment itself or in the behavior of the agents.

## III. LOST AND SO-LOST ALGORITHMS REVIEW

In this section we present a brief review of the LOST and SO-LOST algorithms since our proposed method is based on them. To facilitate the presentation, we consider the resource transportation task between a single source and a single sink. In such a scenario, a robot performs two subtasks: picking resources from the food source and dropping them back when it reaches the home location. In the LOST algorithm, these subtasks are referred to as Events. The goal of the LOST algorithm is to find trails between the source and the home locations. Each robot can recognize each event when it reaches it. The algorithm combines an event and its position at this moment according to the robots local coordinate system in a tuple called Place.

In LOST, robots lay logical waypoints called crumbs imitating pheromones laid by ants in real world. A Crumb is a data structure that consists of the place $P$ to which it refers, a localization space position $L$, a distance estimate $d$ between $L$ and $P$ and finally a time $t$ referring to the creation time of this crumb. The set of crumbs that refer to the same Place is called a trail. Each robot has a temporary local trail which is empty at the beginning. Then, the robot inserts a new crumb to it every fixed time period. This process continues until the robot reaches its goal (either the home or food location). At this moment, the robot broadcasts its temporary trail to all the robots including itself and then deletes the temporary trail. In addition, the robot switches its sub-task and repeats the above process with a new, initially empty, temporary trail.

Each robot maintains a set of crumbs associated with each unique Place mentioned in all received trail messages. When a broadcast from another robot is received, each robot adds the received crumbs to its local set for that Place. LOST algorithm does not assume a global coordinate frame reference between all the robots, so received crumbs are mapped into each robot's local coordinate system before adding them to its local trail. This is done by comparing the coordinates of the common places between all the robots (e.g. the food or home locations) and calculating a unique transformation between the local frames of each robot and other robots. The trails are periodically scanned and crumbs that are older than a threshold are deleted. This imitates the evaporation of real pheromones in real ant trails, and allows the trails to dynamically adapt to the environment.

Based on the above, a robot going to the home location searches the environment within its field of view for all the crumbs that refer to the same location. From those crumbs, the robot chooses the nearest one to the goal location. The output of the algorithm for the robot at this step is (i) the distance that it should travel to reach the chosen crumb and (ii) the heading towards it, that is the angle between the robot's current position and the chosen crumb position. By using the set of crumbs in this way, over time a distinct 'trail' of distance-ordered crumbs emerges between a pair of Places. For efficiency a heap data structure can be used to store the set of crumbs ordered by distance-to-Place.

Although the LOST algorithm shows good performance with a moderate population of robots, this is not always the case with larger robot populations as all robots are attracted to a single best known trail. As the population increases, this can lead to very high robot density and thus increases the probability of spatial interference between robots. This undesirable interference can greatly decrease the overall system performance.

The Spread-Out LOST (SO-LOST) algorithm tackles this problem by ensuring that the trail going to the home location avoids the other trail going to the food source. To achieve this, the Place in any crumb is redefined to refer to the goal location instead of the last visited place (as it is in LOST). The process of laying and following crumbs in SO-LOST is similar to the original process in LOST, but with simple modifications. In the trail following process, the robot in SO-LOST does not only head towards the nearest crumb to the goal location, but it also takes into account the position of crumbs heading to a different goal. If the chosen crumb is close enough to another crumb from the latter category, the robot changes its heading direction slightly to the left. By doing so, robots move in two separate trails; one from home to food and one in the other direction. In [2] the authors show that the system throughput using SO-LOST, in terms of the number of transported goods between the home and

food locations, improves by up to three times more than the LOST algorithm in dense robot populations in constrained environments. However, with large enough populations, the trails will surely saturate. One trail in each direction may not be the best way to distribute robots.

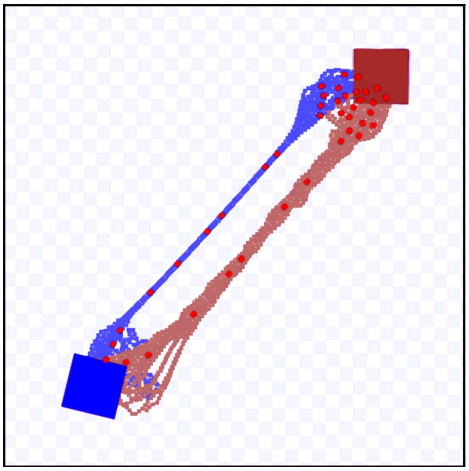
## IV. Proposed Method

The insight of our proposed method is that instead of having only two trails like the case in the SO-LOST algorithm [2], we let the robots construct multiple trails to the same Place. The outcome of our method is having two trails/lanes going from the home to the source and another two going in the opposite direction. Our hypothesis is that using this method, the density of the resultant trails will be reduced. These lanes may be parallel, or may be separated by obstacles. Multiple lanes will increase the overall system performance as when a new lane is added to a congested road in a crowded city.

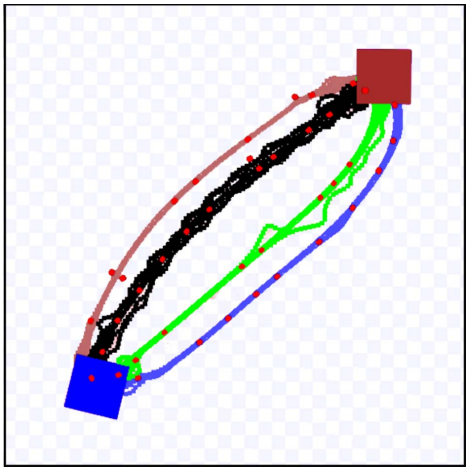
In the spirit of LOST, the method is very simple: we add one more item to the crumb data structure, which is the lane number $l_c$. The lane number refers to the lane at which the robot moves. When a robot lays its first crumb in the environment, it chooses either lane 1 or lane 2 with probability 50%. For the subsequent crumbs, the robot assigns them the same lane number chosen for its first crumb. A robot only follows the crumbs that have both the same place (goal) and lane number. For example, a robot going to the home location in lane 1 only follows the crumbs heading to the same goal and having lane number equals 1, while avoiding other crumbs which we call $c_{anti}$. Therefore, in our case there may be more than one $c_{anti}$ which the robot wants to avoid. For each one of those, the robot checks whether the crumb is in the left or the right of its current position within a distance threshold. If the crumb is in the left (right), then the robot moves its target position slightly to the right (left). It also ensures that the new position does not lie inside an obstacle. This is done by calculating a shift vector for each nearby crumb just like the case in SO-LOST. The sum of all those vectors constitutes the final motion vector that the robot uses in its next move.

We think that our proposed method can lead to several advantages. It can potentially reduce congestion and hence increase the system's throughput in dense robot populations. This is true because the method effectively distributes the robots in the environment and hence reduces the probability of spatial interference.

One interesting observation about our method is that it imitates real ants not only by using the notion of pheromones but also in the way real ants handle congestion. The latter behavior was studied by Couzin and Franks [18], and they conclude that real army ants tend to form lanes while transporting resources in order to minimize traffic congestion.



(a) SO-LOST



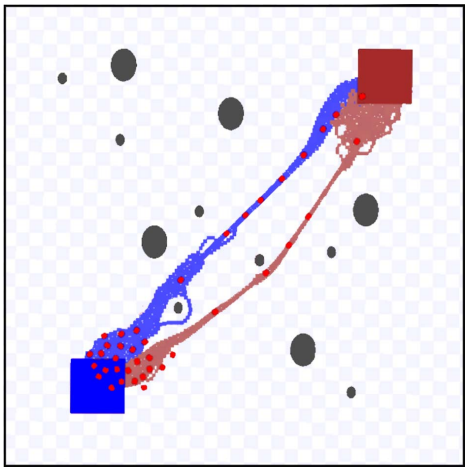
(b) Our Algorithm

Figure 1: The resulting trails in the empty environment using SO-LOST and our algorithm.

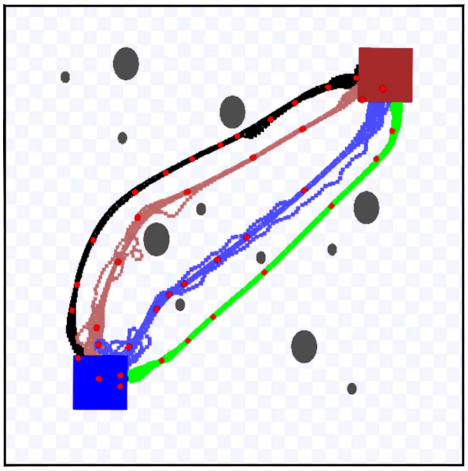
## V. Evaluation

### A. Simulation Setup

In order to evaluate our approach, we use the Stage robot simulator [19]. Our simulated robots are Stage's Pioneer 3DX and SICK LMS200 laser rangefinder models. We run our experiments in three different environments. The first is an empty one (Fig. 1). The second is what we call the dots environment (Fig. 2) where the black dots represent obstacles. The third is the cave environment (Fig. 3). The three environments have the same size of 35x35m. The food source in these environments is the blue square and the home is the red square. Our performance metric is the total number of transported resources between the home and food locations after a period of time. We study how the performance changes with the number of robots. To this end, in each environment we run our experiment 10 times

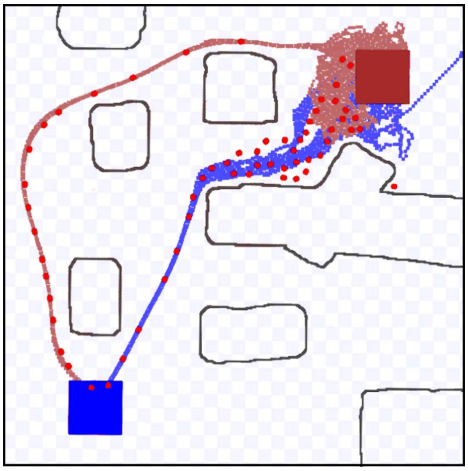


(a) SO-LOST

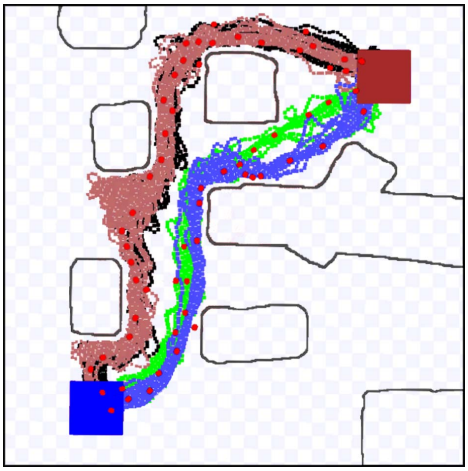


(b) Our Algorithm

Figure 2: The resulting trails in the dots environment using SO-LOST and our algorithm.



(a) SO-LOST



(b) Our Algorithm

Figure 3: The resulting trails in the cave environment using SO-LOST and our algorithm.

for every population size. Each experiment takes two hours. At the beginning of each experiment, robots start moving from the same randomly chosen positions without knowing the position of the home or the food.

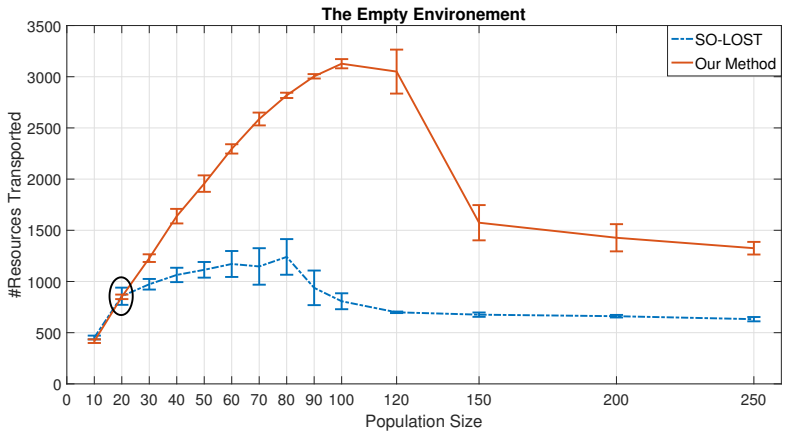
### *B. Results*

Fig. 4 shows the results of our approach. We compare the results of the original SO-LOST [2] with our modified multi-lane method. The empty environment is used to examine how the method distributes robots in the absence of obstacles, where only inter-robot interference affects navigation performance. The dots environment is used to test how our method deals with obstacles. In addition, the cave environment tests our algorithm in limited spaces that contain obstacles. The curves in Fig. 4 represent the mean values of all the experiments and the vertical bars show the standard deviation for each population size.

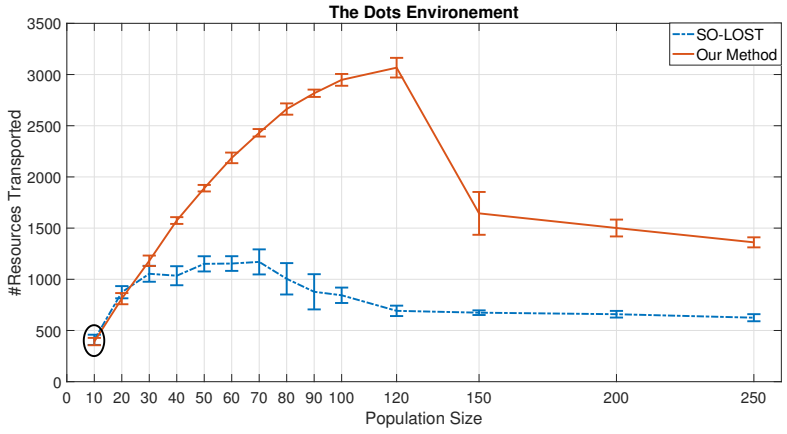
We found the performance of our algorithm is similar to SO-LOST for small population sizes (10 and 20 robots). The reason for this is that the new lanes added in our method are longer than the two trails formed using the SO-LOST algorithm. Moreover, congestion does not usually occur with smaller number of robots and hence it does not challenge the SO-LOST or show the merits of our approach.

Starting from 30-robot population size, our algorithm outperforms the SO-LOST in all three environments. Compared with SO-LOST, our algorithm improves the overall system performance by up to 3.9 times in the empty and dots environments.

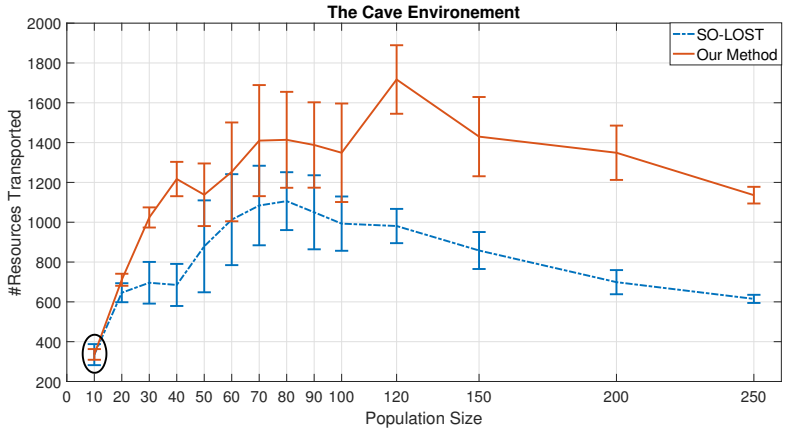
To stress test our method, we perform experiments with even larger population sizes (e.g., 150, 200 and 250 robots). We notice that the system throughput using our method starts decreasing after 100 and 120 population sizes in the empty and dots environments, respectively. Using the SO-




(a)




(b)




(c)

Figure 4: The results of our algorithm compared with the SO-LOST in 3 different environments.

LOST, a similar decrease occurs but with smaller numbers of robots, i.e., after 80 and 70 robots in the empty and the dots environments, respectively. A similar trend occurs in the cave environment where the curve of our method stops fluctuating and decreases gradually after 120 robots. This decrease occurs after 80 robots using the SO-LOST. More-

over, even in these very large population sizes (greater than 120 robots), our method maintains its out-performance and increases the system throughput by up to 133%, 144% and 93% in the empty, dots and cave environments, respectively, compared to the SO-LOST algorithm.

In addition, we notice that the cave environment is the most challenging one for both SO-LOST and our algorithm. This is because it contains larger obstacles and has smaller area for the robots to move. This makes it more difficult for our algorithm because it needs more space to form the lanes. Despite that, our results demonstrate that except for the case of having 10 robots, our algorithm is better than the SO-LOST and it achieves up to 93% increase in the system performance.

We perform hypothesis testing using a T-test to verify that the results of both the SO-LOST and our algorithm are significantly different for each population size. The test shows that for all population sizes, the results of the two algorithms are significantly different ($P << 0.05$). The only exception cases are highlighted in Fig. 4 using ellipses around them. Notice that these cases occur only in small population sizes.

## VI. Discussion

The core idea of our algorithm is that the poor effects of traffic congestion in LOST-type methods can be addressed by better distribution of the robots through the workspace. We take an approach that is sympathetic to LOST by using the notion of lanes from traffic engineering. One of the advantages of our method is that it indirectly addresses the problem of congestion at target locations (i.e. food and home locations). In our method, there are multiple possible entrances to the target instead of only one entrance per direction in the SO-LOST.

We notice that in limited spaces like the case in the cave environment, intersections between different lanes in our method may often occur. However, our algorithm shows improvement in these cases compared with SO-LOST. We observe that in these challenging cases, our algorithm retains the ability of LOST to adapt to regions of high interference, and effectively tends to form wider lanes than usual. Moreover, the intersections occur only between lanes going towards the same goal. These two reasons can justify why our algorithm outperforms the SO-LOST even in these difficult scenarios.

## VII. Conclusion and Future Work

In this paper, we investigated the problem of spatial interference between robots in dense robot populations. We proposed a simple but effective modification of an ant-trail algorithm to use multiple lanes, as utilized in human traffic engineering and large ant populations. Each robot sticks to its lane, and thus avoids spatial interference with robots in other lanes. To test the method's effectiveness at the canonical resource transportation task, we performed several simulation experiments using our algorithm in environments with and without obstacles. Our robots were able to transport up to 3.9 times more resources compared to the baseline single-lane method.

Our results in this paper showed that our algorithm performs much better with large number of robots. In future work, we are interested also in studying the correlation between the size of the environment and the overall system performance. We also demonstrated that by adding one more heuristic (i.e. the lane number) to the crumb data structure the system performance increased significantly. This opens the door to think of more heuristics that can improve the overall performance. Finally, we examined the use of exactly two lanes, yet the method generalizes without modification to any number of lanes. An obvious but possibly very useful extension is to let the system itself dynamically discover an ideal number of lanes based on the current congestion in the environment.

## References


[1] R. T. Vaughan, K. Stoy, G. S. Sukhatme, and M. J. Mataric, "Lost: Localization-space trails for robot teams," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 796–812, 2002.

[2] S. A. Sadat and R. T. Vaughan, "SO-LOST: An Ant-Trail Algorithm for Multi-Robot Navigation with Active Interference Reduction," in *Proc of International Conference on Artificial Life (ALIFE'10)*, Odense, Denmark, August 2010, pp. 687–693.

[3] ——, "Blinkered LOST: Restricting sensor field of view can improve scalability in emergent multi-robot trail following," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA'10)*. Anchorage, Alaska: IEEE, May 2010, pp. 947–952.

[4] A. A. Khaliq and A. Saffiotti, "Stigmergy at work: Planning and navigation for a service robot on an RFID floor," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA'15)*. Seattle, WA: IEEE, May 2015, pp. 1085–1092.

[5] D. Kurabayashi *et al.*, "Realization of an artificial pheromone system in random data carriers using RFID tags for autonomous navigation," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA'09)*. Kobe, Japan: IEEE, May 2009, pp. 2288–2293.

[6] M. Mamei and F. Zambonelli, "Physical deployment of digital pheromones through RFID technology," in *Proc. of IEEE Swarm Intelligence Symposium (SIS'05)*. Pasadena, CA: IEEE, June 2005, pp. 281–288.

[7] R. Mayet, J. Roberz, T. Schmickl, and K. Crailsheim, "Antbots: A feasible visual emulation of pheromone trails for swarm robots," in *Proc. of International Conference on Swarm Intelligence*. Beijing, China: Springer, June 2010, pp. 84–94.

[8] M. Trincavelli, "Gas discrimination for mobile robots," *KI-Künstliche Intelligenz*, vol. 25, no. 4, pp. 351–354, 2011.

[9] R. Fujisawa, H. Imamura, T. Hashimoto, and F. Matsuno, "Communication using pheromone field for multiple robots," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08)*. Nice, France: IEEE, September 2008, pp. 1391–1396.

[10] N. Hoff, R. Wood, and R. Nagpal, "Distributed colony-level algorithm switching for robot swarm foraging," in *Distributed Autonomous Robotic Systems*. Springer, 2013, pp. 417–430.

[11] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee, "Pheromone Robotics," *Autonomous Robots*, vol. 11, no. 3, pp. 319–324, 2001.

[12] F. Ducatelle, G. A. Di Caro, A. Förster, M. Bonani, M. Dorigo, S. Magnenat, F. Mondada, R. OGrady, C. Pinciroli, P. Rétornaz, *et al.*, "Cooperative navigation in robotic swarms," *Swarm Intelligence*, vol. 8, no. 1, pp. 1–33, 2014.

[13] K. Lerman and A. Galstyan, "Mathematical model of foraging in a group of robots: Effect of interference," *Autonomous Robots*, vol. 13, no. 2, pp. 127–141, 2002.

[14] G. Zhou, F. Bastani, W. Zhu, and I. L. Yen, "A Self-Stabilizing Algorithm for the Foraging Problem in Swarm Robotic Systems," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'16)*. Daejeon, Korea: IEEE, October 2016, pp. 2907–2912.

[15] G. Pini, A. Brutschy, C. Pinciroli, M. Dorigo, and M. Birattari, "Autonomous task partitioning in robot foraging: an approach based on cost estimation," *Adaptive behavior*, vol. 21, no. 2, pp. 118–136, 2013.

[16] A. Brutschy, G. Pini, C. Pinciroli, M. Birattari, and M. Dorigo, "Self-organized task allocation to sequentially interdependent tasks in swarm robotics," *Autonomous agents and multi-agent systems*, vol. 28, no. 1, pp. 101–125, 2014.

[17] A. Scheidler, D. Merkle, and M. Middendorf, "Congestion control in ant like moving agent systems," in *Biologically-Inspired Collaborative Computing*. Springer, 2008, pp. 33–43.

[18] I. D. Couzin and N. R. Franks, "Self-organized lane formation and optimized traffic flow in army ants," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 270, no. 1511, pp. 139–146, 2003.

[19] R. Vaughan, "Massively multi-robot simulation in Stage," *Swarm Intelligence*, vol. 2, no. 2-4, pp. 189–208, 2008.