SO-LOST: An Ant-Trail Algorithm for Multi-Robot Navigation with Active Interference Reduction

Seyed Abbas Sadat, Richard T. Vaughan

Autonomy Lab, School of Computing Science
Simon Fraser University, BC, Canada
{*sas21, vaughan*}*@sfu.ca*

Abstract

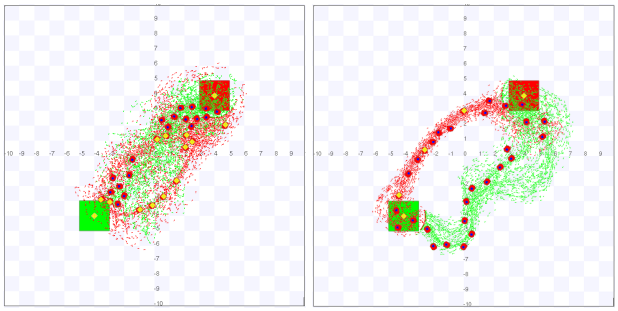
We consider a group of autonomous robots which perform the classical task of transporting resources from a source to home. The robots use ant-like emergent trail following to navigate between home and source. When trails lie close together, spatial interference between robots navigating in opposite directions reduces overall system performance. This paper proposes a navigation strategy which is effective in separating trails with different goals. The results of simulation experiments indicate that the performance of robots is usefully increased compared to original algorithm in constrained environments.

Introduction

This paper presents a navigation strategy to reduce interference in ant-inspired foraging-and-trail-following robot systems. We consider the classical *resource transportation* task, in which a team of robots works to transport resources in an initially unmapped environment. Robots start from a home position and search for a supply of resources. On reaching the source, they receive a unit of resource and must return home with it, then return to fetch more resource repeatedly for the length of a trial. Achieving this task reliably with robots will meet a real-world need. It is a canonical multi-robot task since the work is inherently parallelizable. The critical factor limiting scalability is mutual spatial interference between robots.

Our earlier work Vaughan et al. (2000, 2002) examined an implementation of ant-inspired trail following that is suitable for imperfectly-localized mobile robots. In our "localization-space trails" (LOST) algorithm, robots generate and share trail data structures composed of waypoints specified by reference to task-level features that are shared by all robots. The trails are continuously refined online, and maintain the ant-algorithm property Dorigo (1992) of converging to near-optimal paths from source to home.

Trails are labelled with their destination, and the trail to the current goal destination is followed. In previous work, the other trails were ignored during navigation. However, trails may overlap in space and robots navigating to different goals may interfere with each other's progress. We argued previously that an emergent property of LOST is that it can produce trails that are separated in space Vaughan et al. (2000) thus reducing interference. In this paper we describe a modification to LOST called Spread-Out LOST (SO-LOST) that greatly improves this effect, creating trails that share parts of the environment while being far enough apart to reduce interference. The result is superior performance in most of the cases we examine. The innovation is that the robots' trail-following behaviour is subtly modified to avoid competing trails, with the emergent effect that trails are iteratively spread out until intereference is largely avoided.



(a) LOST (b) SO-LOST

Figure 1: Trails formed in an obstacle-free "empty" environment using LOST and SO-LOST. SO-LOST has separated the trails, achieving better throughput due to reduced interference.

It is reported that some type of ant use repellent pheromone to mark unrewarding areas so that other ants avoid foraging that part of the environment (Robinson et al. (2008)). However, we do not know of any biological system that uses similar behaviour to tackle the spatial interference problem. The new navigation algorithm in this paper is a synthetic technique that improves the efficiency of a biologically-inspired path finding and sharing algorithm used in multi-robot systems. The advantage of these type of synthetic behaviors has been studied before (e.g. Heck and Ghosh (2002)).

Related Work

Various different robot implementations of ant-like trail following have been presented. Real chemical marks were first used to produce true stigmergic trail-following in Russell et al. (1994). Also recently, Fujisawa et al. Fujisawa et al. (2008) carried out a study out of communication in a swarm of robots using pheromone and proposed a behavior algorithm for robots to search for prey and attract other robots. They used ethanol as pheromone in their real robot experiments. The challenge of chemical and sensor engineering makes these methods often impractical. A more parsimonious method was invented by Payton et al. (2001) where virtual pheromone trails are implemented by directional infra-red messages transmitted from robot to robot. Robots echo received messages, incrementing a contained hop-count which is used to estimate the distance to the message source. In both chemical and IR-mediated methods, the local "gradient" is sensed directly from the environment. If robots are mutually localized, virtual trails can be created from global waypoints, which are communicated by wireless network. We showed that this scheme can be robust to large zero-mean localization error (Vaughan et al. (2000)), and admits a relaxed and practical definition of mutual localization (Vaughan et al. (2002)).

The diminishing-to-negative-returns effect of increasing the number of robots on performance has been studied in related contexts. In a mathematical model of robot foraging Lerman and Galstyan (2002), it was shown that adding more robots to the system improved the group performance while decreasing individual robot's performance. Based on that model, an optimal group size was found that maximizes the group performance. Explicit anti-interference strategies are studied in real robots in Zuluaga and Vaughan (2005), to increase performance in the transportation task. Congestion control in a dense multi-robot system is studied in Scheidler et al. (2008), where asymmetries that resolve conflicts are introduced by modifying either the environment or the robot controllers.

A related idea using occupancy grids to model multi-robot interaction is described in Zuluaga and Vaughan (2008). There, a global histogram of occupancy is constructed, and areas with high probability of co-location are identified and fed into an (unrelated) interference reduction method.

Localization-Space Trails (LOST) review

This section briefly reviews the generalized trail-following method formulated in Vaughan et al. (2002).

LOST generates trails between the locations of *Events*. An Event is defined as a task-relevant occurrence that may happen to any member of the team, and is locally but reliably perceived. For example, in our transportation task the relevant Events would be 'pick-up-resource' and 'drop-resource'. A robot must be able to recognize these events in order to switch between resource-seeking behavior and home-seeking behavior. When an Event occurs to a robot, its current pose in localization space is recorded to create an [Event,Pose] tuple called a *Place*. A robot can then express information about the world relative to the Places it has seen. Other robots that have position estimates for the same Events can interpret the coordinates in their own local frame of reference. Thus robots are mutually localized by the shared experience of the common task, rather than conventional global localization in some arbitrary coordinate system.

The purpose of LOST is to guide the robot to a Place currently of interest: the goal. The algorithm provides the robot controller with two pieces of information; (i) the *heading-hint* that is the local direction in which to travel to reach the goal; (ii) the *distance-hint* that is the estimated cost (usually in time) to reach the goal. These hints are extracted by examining a set of waypoints called *Crumbs* which are poses specified relative to a Place. The current set of Crumbs specified relative to a particular Place is a *Trail* to that place. A Crumb is a tuple $C = [P_c, L_c, d_c, t_c]$ containing the name of the Place P_c to which it refers, a localization space pose L_c, an estimate d_c of the distance (in some distance function) from L_c to P_c, and the time t_c when the Crumb was created.

Each robot maintains an initially empty temporary trail. Every S seconds, a robot inserts a new crumb to the temporary trail. The crumb contains the current location of the robot, the name of the most recent Event experienced by that robot, the distance from the last event, and the current time. When another event occurs to the robot (e.g., when a robot drops off its cargo), the temporary trail is broadcast to all robots, including itself, then deleted. A new temporary trail is then created for the recent Event.

Besides the temporary trail, each robot maintains a trail for each different Event it has learned about from the network. When a broadcast trail is received, the crumb poses are transformed into the local frame of reference by the rigid body transform defined by comparing the local and received poses of the trail's Place. The transformed crumbs are added to the local trail for this Place. All trails are periodically scanned and any Crumb with time stamp older than age threshold a seconds is discarded. Thus the trail is updated dynamically, and out-of-date information is expired. The dynamic response of the trail to changing environments is a function of a.

Suppose a robot at pose L_r has Place P_g as its goal, such as $Event(P_g)$ = 'drop-resource-at-home'. The robot searches the set of Crumbs with Place = P_g to find the set of crumbs that lie within its *field of view* (FOV) i.e., within radius d_f of L_r. From this set it finds the crumb C_L with the smallest distance-to-goal d_c. This distance is returned as the distance-hint. The heading-hint is the angle from the robot's pose L_r to $L_c = Pose(C_L)$. If the robot moves in the direction of the heading hint and repeats this process, it will encounter crumbs with decreasing distance to goal values,

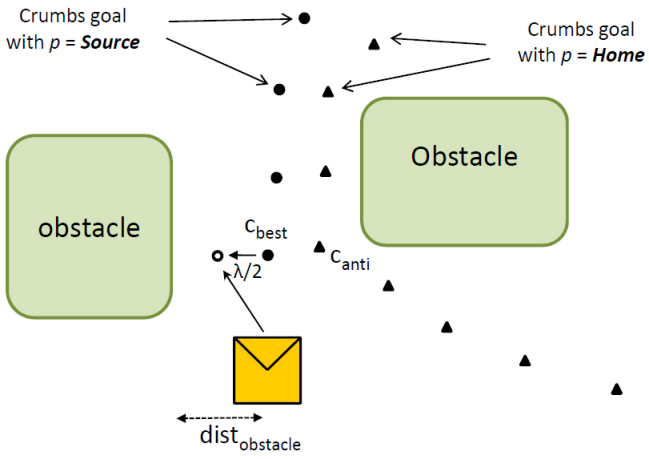


Figure 2: Sketch of the new LOST algorithm. While the robot was following the trail (filled circles), it sees a crumb with different goal (triangle) and thus changes its direction to a new point (the empty circle).

and eventually arrive at P_g.

The robot will take the shortest route so far discovered from that location. By following the Crumbs dropped by the whole population, each robot benefits from the others' exploration; robots will find a reasonable route much more quickly than they would alone. The larger the population size, the greater the probability of finding a good route and the more quickly a good route is found.

Spread-Out LOST

In the LOST algorithm, as the robots move they "lay" crumbs. The goal Place of these crumbs is the place that the robot has most recently visited. This means that in order to reinforce a trail, the robots should travel in the opposite direction that the crumbs are showing and consequently robots following a trail are very likely to interfere with robots laying (reinforcing) it. With few robots, this does not have much effect on performance and the "pick-up-resource" and "drop-resource" trails converge to one shortest discovered path. However as the robots' team size increases, these interferences damage the performance of the system.

To address this, we modify the LOST algorithm so that when a crumb is created, the P_c data field will be the goal of the robot rather than the recently visited place. With this modification, the robots have to perform two searches at the beginning; one for finding a path from home to source and another one for a path from source to home. We can avoid the need for the second search by copying the first discovered trail and changing the goal and reversing the distance hint along the trail.

When the environment in which the robots are working

Algorithm 1 The New Trail-Using Algorithm

Require: The distance $dist_{obstacle}$ from the robot to the nearest non-robot obstacle on the left side of the robot.
return the direction Dir_{robot} to which the robot should move

Θ = all the crumbs in the robot's FOV with positions relative to the robot;
$\Sigma = \{c | c \in \Theta \wedge (c.p_c = robot.goal)\}$;
$\Pi = \{c | c \in \Theta \wedge (c.p_c \neq robot.goal)\}$;

$\lambda = Min(crumb\_avoid, dist_{obstacle})$;

$c_{best} = c$ s.t. $(c \in \Sigma) \wedge (\nexists c' \in \Sigma$ s.t. $c.d_c > c'.d_c)$;

if $(\exists c_{anti} \in \Pi$ s.t. $dist(c_{anti}, robot) < crumb\_avoid) \wedge (c_{best}.d_c \leq 2s)$ **then**

$Dir_{robot} = \overrightarrow{(robot, c)} + \frac{\lambda}{2} \times \overrightarrow{(-1, 0)}$;

else

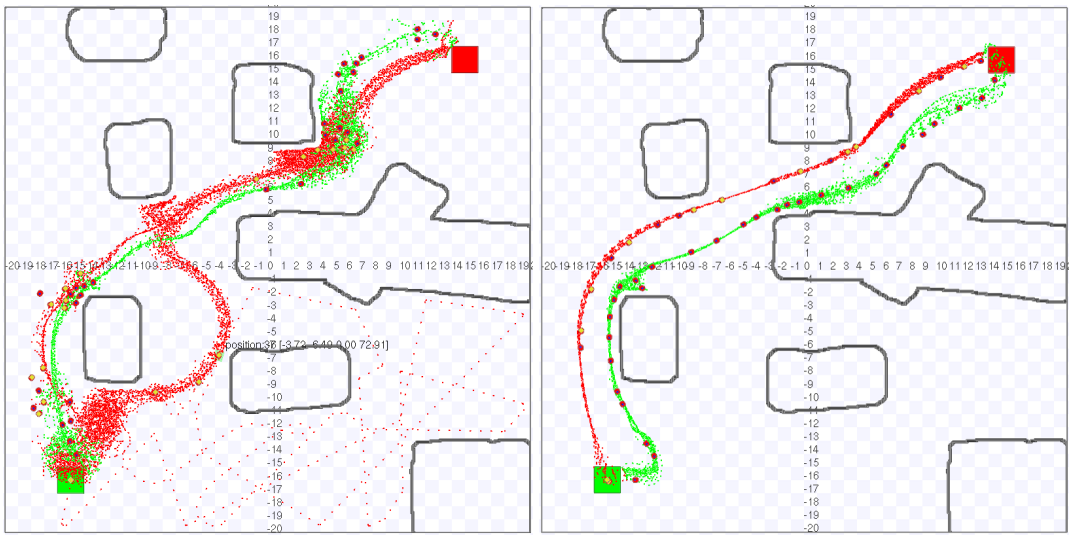
$Dir_{robot} = \overrightarrow{(robot, c)}$;

end if

is complicated and contains narrow corridors and doorways, or is very crowded, LOST may produce trails with different goals that are either very similar or have many parts in common. Figures 1(a),3(a),4(a) show this phenomenon in our trail-following robot system implemented in the well known simulator Stage (Vaughan (2008)). The trails formed between source and home are often very close to each other, leading to problematic interference between robots travelling in opposite directions. Since the crumb trail data structure does not contain any explicit information about the fixed obstacles in the environment, there is no way to directly process the trail data to avoid robot-robot interference without risking directing robots into fixed obstacles. Instead, we use a small modification to the robots' trail following control strategy that results in emergent trail separation.

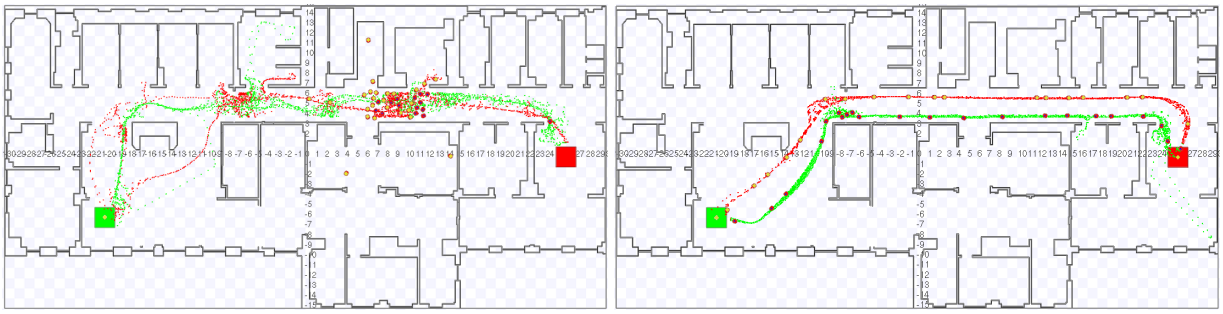
A robot following a trail to get to P_c, can interpret crumbs with goals other than P_c, as proxies for potentially interfering robots. If the robot follows the trail to P_c while slightly avoiding all other nearby crumbs, the new P_c crumbs it lays will tend to be slightly more distant from other crumbs than those just followed. This mechanism is essentially similar to the iterated corner-cutting that drives the ant-algorithm's ability to locally improve trail length. The resulting trails may be slightly longer but may reduce interference significantly, as suggested by the results below.

The new trail-using algorithm is presented in Algorithm 1. It first searches for the crumb c_{best} with minimum distance to goal that is located in the robot's FOV. Then if



(a) LOST (b) SO-LOST

Figure 3: Trails formed in the cave environment using the LOST and the new algorithm after 30 mins of simulation.



(a) LOST (b) SO-LOST

Figure 4: Trails formed in the hospital environment. using the LOST and the new algorithm.

there exists a crumb c_{anti} with different goal than the robot's goal and it was closer to the robot than a distance threshold ($crumb\_avoid$), the direction to which the robot moves will turn to the robot's left. This will change the angular velocity of the robot so that it keeps away from c_{anti}. The shift vector is orthogonal to the $\overrightarrow{(robot, c_{best})}$ vector. Also, the magnitude λ is calculated based on the obstacles near the robot such that the robot's target point does not lie inside an obstacle. Trails with different goals are necessarily very close to each other around source and home. Thus the shift vector is not applied when the robot is near the goals to prevent robot's circular trajectory in these areas.

Figure 2 illustrates how the behavior of the robot changes in presence of c_{anti}. The robot is following the small circles. On seeing the triangle crumbs, the robot's target point is changed from c_{best} to another point (the empty circle). This simple mechanism alters the robots movement so that different trails are gradually separated from each other. The divergent movement of trails continues until they are away enough from each other, if possible.

Experiments

Simulation Setup

We ran Stage simulations to evaluate the new algorithm in three different environment settings: $empty$ (Figure 1), $cave$ (Figure 3) and $hospital$ (Figure 4). The size of the $empty$, $cave$ and $hospital$ environments are 20x20m, 40x40m and 60x30m respectively, with robot length 0.45m. Robots are Stage's Pioneer 3DX and SICK LMS200 laser rangefinder models. The bottom left (green) square is the source; top right (red) square the sink of resources. In the screenshots, robots (red polygons) are shown with yellow diamonds to indicate they are carrying a unit of resource. Robots start every trial at the same randomly-chosen uniformly distributed positions, do not know the initial location of source and sink locations, and must find them by exploration at the start of

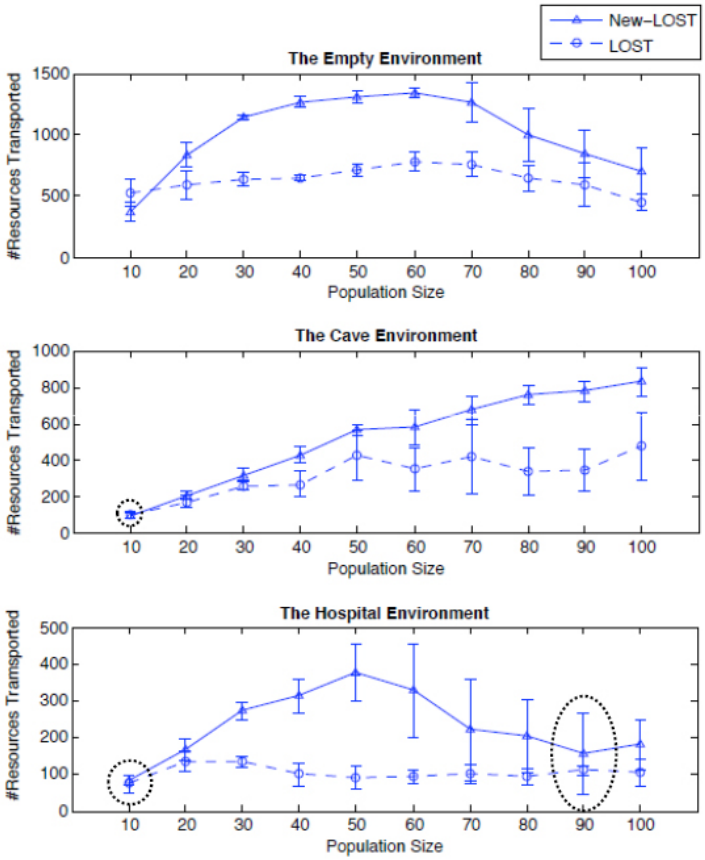


Figure 5: The result of the experiments in the 3 environments. The mean performance over 10 trials are shown with errorbars showing the standard deviation for both the original LOST and the new algorithm. The dotted line shows the data point for which the two algorithm do not show significant difference in distribution.

the trial. Each trial runs for 60 minutes, and the total number of resources delivered at the end of the trial is our performance metric. 10 trials are performed for each population size. LOST is deterministic but the local obstacle avoidance and searching is stochastic (for robustness), hence the need for repeated trials. For all experiments the $crumb\_avoid$ parameter is set to 2m.

Results

The results of the experiments are summarized in Figure 5, showing the mean and standard deviation of performance over 10 repeated trials plotted for each population size. The plot shows a marked improvement in many cases (in some cases 3 times better) in performance with the new algorithm.

As expected, with few robots (20), there is not much difference in performance since the interference among robots is small. In the empty environment with population size of 10, the LOST outperforms the new algorithm. This is because the benefit of interference reduction can not outweigh the penalty of increase in the length of the trails. As the population size increases and the environment becomes more constrained, improvement in performance gets bigger. This can be seen in the plot showing the results of the experiments in the hospital environment; For the smallest populations, the two methods perform about the same; however, since the hospital environment contains corridors and doorways (Figure 4(b)), there is a degradation in the LOST performance with more robots whereas the new algorithm improves the performance in some populations up to 3 times.

To verify that the performance results are significantly different for different algorithms, we performed hypothesis testing using a T-test. The P values for the hypothesis that the performance values for LOST and the new algorithm are from the same distribution are calcuated. For all population sizes, the test suggests that the distributions are significantly different ($P << 0.02$), except for the pairs identified in Figure 5 with dotted line.

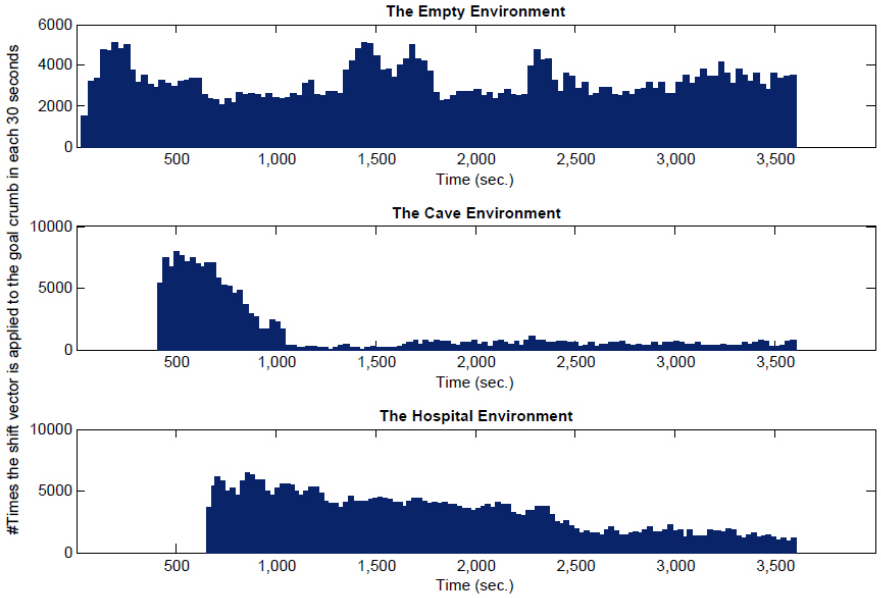


Figure 6: The histograms showing the number of times the goal crumb was shifted. The time bin is 30 seconds of simulation time. 30 robots are used in the empty environment and 50 robots are used in the other environments.

Discussion

The new algorithm is based on the idea that laying crumbs near other crumbs with different goals increases the probability of co-location among the robots performing different tasks. This is more clear in transportation task in which the trails for 'pick-up-resource' and 'drop-resource' tasks can be formed very close to each other. In the new algorithm robots follow the trails and also try to keep a distance from other crumbs and therefore new trails are laid at a safe distance from each other. Figures 1(b), 3(b), 4(b) show the trails formed with the new algorithm. It is visible that different trails are separated from each other and consequently robots do not approach the unattractive trails. The magnitude of the shift vector ($crumb\_avoid$) determines the distance of the trails from each other and should be large enough to keep robots away from each other.

In order to see if the trails converge to a stable state we plotted the number of simulation cycles in which the shift vector was applied in each 30 sec of simulation time (Figure 6). In the $cave$ and $hospital$ environments, after the trails are formed they are gradually separated from each other due to the high use of shift vector. After some time, the trails come into a relatively stable state. The shift vector is still applied occasionally since the trails in some narrow parts of the environment (like doorways) are at their maximum distance from each other and can not go farther away. For the $empty$ environment since the area is small and there is a short distance between source and sink, the robots tend to be pushed towards other trails which results in the high use of shift vector throughout the experiment.

We do not know of any biological system that uses a similar approach to reduce destructive effects of interference among individuals, but still we believe that these techniques can be used in systems inspired from animals and social insects to improve the efficiency of robots in performing a task.

Conclusion and Future Works

In this paper we presented SO-LOST, a new navigation strategy to reduce interference in ant-inspired foraging-and-trail-following robot systems. The method makes use of the different trails formed in the environment to prevent robots with different goals from getting in each other's way. It is quantitatively evaluated through simulation experiments and shown to be effective in relatively constrained environments. Qualitatively, the screenshots of simulation experiments show that distinct separate trails with different goals were formed while keeping a distance from each other hence reducing the interference.

In future work we will implement the new algorithm on real robots and run experiments to verify our findings in simulation. Also, we will investigate methods of congestion resolution in trail-following robot systems. The algorithm presented in this paper is used to avoid congestion and conflicts between robots. However, there is plenty of room for improvement in mutual robot-robot avoidance methods, and development here would have an impact in many multi-robot systems.

The LOST and SO-LOST framework allows us to add various kinds of meta-data to the crumb and trail data structures. Here we have allowed all nearby trails to influence the behaviour of a trail-follower. We expect that performance could be further improved by clever use of other meta-data embedded into crumbs, perhaps by gathering some global statistics. This would be unusual in ant-inspired systems, and perhaps powerful.

For now, we believe SO-LOST may be the most real-world practical trail-following algorithm yet described, since it explicitly manages the spatial interference that plagues real-world robots in any number.

Acknowledgements

This work was supported by NSERC and DRDC in Canada. Thanks to Greg Broten at DRDC, and to members of the Autonomy Lab for their constructive input.

Code publication

All source code, scripts, etc. used to produce the results reported in this paper are available online:
URI: http://www.cs.sfu.ca/ sas21/personal/abbas_alifeXII.tar.gz
SHA1: 9ea88a34d9971dffce26bc511f6ad2f875b8e44d

References

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano, Italy.

Fujisawa, R., Imamura, H., Hashimoto, T., and Matsuno, F. (2008). Communication using pheromone field for multiple robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1391–1396.

Heck, P. S. and Ghosh, S. (2002). The design and role of synthetic creative traits in artificial ant colonies. *J. Intell. Robotics Syst.*, 33(4):343–370.

Lerman, K. and Galstyan, A. (2002). Mathematical model of foraging in a group of robots: Effect of interference. *Auton. Robots*, 13(2):127–141.

Payton, D., Daily, M., Estowski, R., Howard, M., and Lee, C. (2001). Pheromone robotics. *Auton. Robots*, 11(3):319–324.

Robinson, E. J., Ratnieks, F. L., and Holcombe, M. (2008). An agent-based model to investigate the roles of attractive and repellent pheromones in ant decision making during foraging. *Journal of Theoretical Biology*, 255(2):250 – 258.

Russell, A., Thiel, D., Deveza, R., and Mackay-Sim, A. (1994). Sensing odour trails for mobile robot navigation. In *Proc. Int. Conf Robotics and Automation*, pages 2672–2677. IEEE.

Scheidler, A., Merkle, D., and Middendorf, M. (2008). Congestion control in ant like moving agent systems. In *Biologically-Inspired Collaborative Computing*, volume 268, pages 33–43.

Vaughan, R. T. (2008). Massively multi-robot simulations in stage. *Swarm Intelligence*, 2(2-4):189–208.

Vaughan, R. T., Støy, K., Howard, A., Sukhatme, G., and Matarić, M. J. (2002). Lost: Localization-space trails for robot teams. *IEEE Transactions on Robotics and Autonomous Systems*, 18(5):796–812.

Vaughan, R. T., Støy, K., Sukhatme, G. S., and Matarić, M. J. (2000). Whistling in the dark: cooperative trail following in uncertain localization space. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 187–194, Barcelona, Spain.

Zuluaga, M. and Vaughan, R. (2005). Reducing spatial interference in robot teams by local-investment aggression. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Alberta.

Zuluaga, M. and Vaughan, R. T. (2008). Modeling multi-robot interaction using generalized occupancy grids, with application to reducing spatial interference. In *Proceedings of the IEEE International Conference on Robotics and Automation*.