An Ant Algorithm for Multi-Robot Navigation and Distributed Simultaneous Task Allocation

Faraz Shamshirdar, Rakesh Shrestha and Richard Vaughan

Abstract— **This paper describes multi-robot foraging in an unknown environment with an unknown number of sources and a single sink (home). The goal is to effectively allocate robots to sources with least interference and maximum rate of resource collection. The navigation between source and sink is achieved by using trails of virtual landmarks inspired from ant-like trail laying behaviour.**

The algorithm is tested in simulation with varying number of robots and sources. The result indicate that our method achieves near optimal allocation of robots to different sources.

I. INTRODUCTION

We present an algorithm for the classical *resource transportation* task in an unmapped environment. Robots start from arbitrary positions and search for a supply of resources. On reaching the source, they receive a unit of resource and return home with it, then return to fetch more resource repeatedly for the length of a trial [**?**]. Our main focus in this paper is finding task allocation that maximizes the rate of collection of the resources (*throughput*) and with minimum congestion in trails. These two goals are mostly overlapping, hence we achieve the former by trying to solve the latter.

The earlier works [**?**] for this task present Localization-Space Trails (LOST) which is an implementation of ant-inspired trail following for imperfectly-localized robots. In that algorithm, robots generate and share trail data structures composed of waypoints specified by reference to task-level features. The trails are continuously refined online, and maintain the ant-algorithm property [**?**]DORIGO1992) of converging to near-optimal paths from source to home.

In the LOST and other ant-algorithm based methods, time is used as the distance function to be optimized, by which the discovered paths can be longer in space but shorter in time since they spread robots out to minimize spatial interference between robots. By increasing the population size, these interferences eventually dominate the traveling cost. However, since ant algorithms tend to converge to a single best trail, in a large population this train can become congested. Since the behaviour of the ant algorithms (including LOST) is to explore the environment to discover the target while finding the shortest path to that, in the case of multiple sources – which we are trying to explore – they do not perform well since they try to minimize the path to a random source (which might be the closest one). Hence, we modify the original algorithm such that it does not converge to a single source as it would result in better throughput.

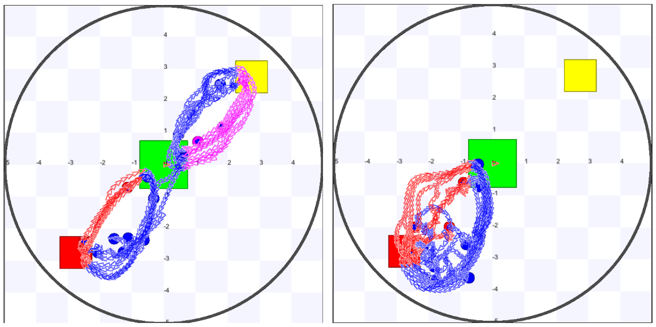

Fig. 1: Trails formed in an empty environment using our embodied apprach outlined in section IV (left) and SO-LOST (right)

In this paper we describe two modifications to the existing LOST algorithm that improve performance in the case of multiple sources and a large number of robots as well as maintaining the same performance in a single-source, single-sink setting with less number of robots. We were able to get superior performance in most of the test-cases we examined.

II. RELATED WORKS

Various works of ant-inspired trail following algorithms have been published, ranging from leaving real chemical trails [**?**] to infra-red message transmission [**?**].

Our work closely follows from Localization-Space Trails (LOST) [**?**] and its variants which is suitable for imperfectly localized robots. A brief review of LOST will be provided in subsection II-A. The method is to generate and share short-lived waypoints (*crumbs)* by robots, which contain some information (estimated distance to target, time to target, density of waypoints, to-source or to-sink label). By following these crumbs, robots can find optimal path between source and home. In the Spread-Out LOST (SO-LOST) which is a modified version of LOST, it tries to increase the scalability by minimizing mutual spatial inference between robots which is done by shifting crumbs in a trail from source to home and home to source for a single source, single sink.

the Multi-Objective LOST (MO-LOST) tries to reduce mutual spatial inference for paths with multiple sources, multiple sinks by shifting the crumbs of a trail which is crossing other crumbs of the other trails. In the Blinkered LOST (Blinkered LOST) it tries to modulate the field of the view (FOV) of the robots' trail-detecting sensor. it shows that the global throughput is a function of robot FOV, means that with narrower FOVs, it performs better in large populations, since narrrower FOV causes multiple trails to be maintained,

Autonomy Lab, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada

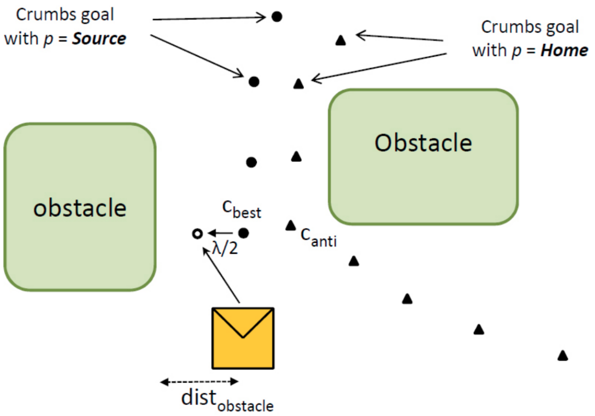

Fig. 2: Trail laying behavior in SO-LOST. The robot is following trail to source (filled circle) and changes its direction to new point (empty circle) to keep distance from trail leading to home (triangles).

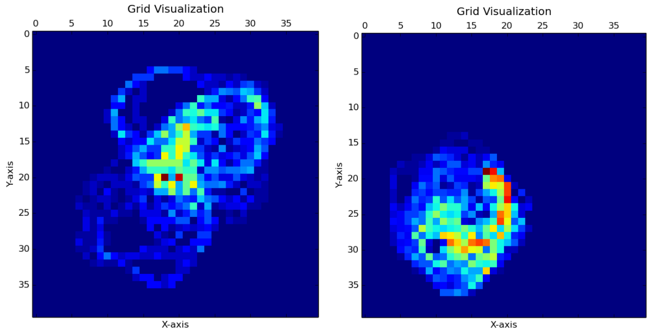

Fig. 3: Grid visualization in embodies approach (left) and original version of SO-LOST (right). The home is at center while the sources are at top-right and bottom-left

so that the system can support larger population sizes before saturating due to inference.

A. LOCALIZATION-SPACE TRAILS REVIEW

LOST generates trails between the locations of *Events* which in our case is reaching source and reaching home. When an Event occurs to robot, its pose in localization space is recorded as *Place* represented by tuple $[Event, Pose]$. Other robots can interpret the coordinates of a place in their own frame of reference. To guide robots between these Events, waypoints called *Crumbs* are laid in every S seconds. A Crumb is a four-tuple $[P_c, L_c, d_c, D_c, t_c]$ containing Place name P_c of place of interest, Localization-space pose L_c, estimated distance (in some distance function) between P_c and L_c, and t_c the Crumb creation time. A series of Crumbs that lead to the place P_c is referred to as a $Trail$. A robot broadcasts its Trail when an even occurs to it (i.e. when it reaches source or home). All trails are periodically scanned and Crumbs with timestamp older than age threshold a seconds are discarded. This is analogous to ant's pheromone trails evaporating over time.

A robot follows trail to a destination place P_g by heading towards the best crumb C_L within its *field of view*, i.e. within radius d_f of its current position. We use the modified version of this algorithm SO-LOST which lays crumbs certain distance λ away from crumbs leading to a different place. Figure 2 shows the crumb laying behavior of the SO-LOST algorithm.

III. CONTRIBUTION

The main contribution of this paper is modification of LOST algorithm to have a simultaneous task allocation which performs better in multiple-source, single-home setting in both large and small population sizes. Two approaches were explored:

1) Embodied approach: Subtle modification to LOST using trail density as a proxy for congestion which has the emergent effect of maintaining multiple trails leading to different sources
2) Explicit allocation approach: Allocate robots to different sources with a goal of uniform distribution of robots along different trails (with consideration of the distances of the sources).

In the following sections, we will describe these approaches with quantitave evaluations.

IV. EMBODIED APPROACH

This approach maintains trails to multiple sources without keeping track of the number of sources discovered so far. The general workings of this approach is same as that of LOST algorithm where the robots try to follow the best crumb in their immediate neighbourhood. The crumbs are not labelled with source identification and the robots still make local decision without explicitly considering the multiple sources available.

The reason LOST and its other variants converge to a single congested trail is that the robots try to minimize the distance to the trail without considering the congestion in the trail. Even when time to target is using as distance metric instead of physical distance, the final trail is still congested for large population sizes.

This conflict here is whether to follow the shortest path discovered so far or avoid following congested trail which potentially leading to the target with shortest distance. To make a better decision, we take both these factors into account and make decision based on both factors. For this, the distance function is formulated as follows:

$$\begin{aligned} d = w_{time\_to\_target} \cdot time\_to\_target + \\ w_{crumb\_density} \cdot crumb\_density \end{aligned} \tag{1}$$

where $time\_to\_target$ is the time to target encoded in the crumbs and $crumb\_density$ local density of trail around a crumb (to be discussed shortly).

The best values of $w_{time\_to\_target}$ and $w_{crumb\_density}$ were found to be 0.3 and 0.7 respectively from experiments[1]. Note

[1] The scaling of the weights does not affect the performance. So experiments were done with weights that summed to one

that when $w_{crumb\_density}$ is set to 0, the algorithm is the same as SO-LOST.

This method not only maintains trails leading to multiple sources, each trail isn't just a single curve of crumbs (Figure 3). Hence, there are multiple side trails that lead to the same source.

A. CRUMB DENSITY IMPLEMENTATION

For finding crumb density, we implemented an additional data structure which is a 2D grid that spans the simulation world. Each cell of the grid contains the number of crumbs within it. Hence, the grid represent a histogram of crumbs. The $crumb\_density$ around a crumb is then the number of crumbs in the cell it belongs to divided by total number of crumbs present. We used cells of size 0.25 x 0.25 m. This is approximation of a robot's size. Figure 3 shows a visualization of our grid data structure. A brighter value means higher number of crumbs. The fact that SO-LOST implementation converges to a single source is clearly visible while our approach maintains trails leading to both the sources.

B. DRAWBACKS OF THIS APPROACH

A major drawbacks of this approach is that there are a lot of parameters that needed to be handtuned. These parameters are evaporation rate of crumbs, grid size and weights of density and time to target in distance function. Even though we were able to get moderate invariance to size of the world, number of robots, number of sources and distances of the sources, these parameters still need fine tuning when used in drastically different setting.

A future extension would be to change these parameters dynamically. For example, we can set evaporation rate based on distance between source and home such that multiple cycles on the same trail does not give a false impression of congestion.

Also, there is also no guarantee of convergence to multiple sources. If by chance all the robots discover the same source, there will only be a single trail leading to that source (although this is highly unlikely if we have large number of robots).

V. EXPLICIT ALLOCATION APPROACH

This method was explored to mitigate the shortcomings of our previous approach. Using crumb density as a heuristic of congestion introduces unwanted dependence on evaporation rate and trail length as discussed in the previous subsection IV-B. This approach, instead of using approximation, utilizes the actual congestion in different trails. It also modifies the crumb data structure to include the source the crumb is leading to and the total euclidean distance from that crumb to the target, so the new crumb data structure with respect to the its definition in the LOST review section is a tuple $C = [P_c, S_c, L_c, d_c, D_c, t_c]$ which contains the name of the Place P_c to which it refers [source or sink], source id that the crumb is leading to S_c, a localization space pose L_c, an estimate d_c of the distance from L_c to the target P_c which comes from a distance function that assume time as distance in the original version of LOST, an estimate of total euclidean distance from the crumb to the target, and the time t_c containing the creation time of the crumb.

We also maintain a global data structure to keep track of numbers of robot currently following different sources. It is implemented as a hash table with all the discovered sources as its key and the number of robots in trail leading to a source as its value. This structure is updated every time a robot reaches home and is allocated to a source.

When a robot reaches home, it checks congestion in every trail. We formulation our *congestion_factor* as follows:

$$congestion\_factor = \frac{(R_{SIZE} + D_{SAFE}) \cdot n_t}{d_t} \quad (2)$$

where R_{SIZE} is the size of robot (diameter) in meters, D_{SAFE} is a safe distance (meters) we need to maintain between the robots, n_t is the number of robots currently in that trail and d_t is the length of the trail (meters)

The ideal $congestion\_factor$ is 1 where the robots form a tight loop in the trail with a safety distance D_{SAFE} between each other. Whenever a robot reaches home, it calculates $congestion\_factor$ for each source and decides to choose the source with minimum $congestion\_factor$ that is less than 1. If no trail is available that has $congestion\_factor$ less than 1, the robot decides not to follow any trail and move randomly to find other food sources. After choosing the source (or deciding to explore randomly), the hash table is updated if needed.

Although this method maintains trail leading to multiple sources and increases our throughput, but in the case of having paths that are almost equally congested, robots can switch back and forth between sources. We introduce an *inertia* factor for convergence of robots to a particular source; a robot decides to switch from one source to another only if the $congestion\_factor$ is better than that of previous source by a fixed threshold.

This approach will have a better performance even in single source setting as the trail between source and home is maintained such that there is minimum change of interference between the robots. Experimental results showing this will be provided in the following section (Section VI). In our implementation, we do not restrict the wandering robots that were not assigned to any known source to cross existing trails. A better performance will likely be achieved by having such restrictions.

A. POSSIBLE MODIFICATIONS FOR DISTRIBUTEDNESS

In the previous section we mentioned a new data structure that used to keep number of robots in each path, which makes this approach centralized, the following defintion of crumbs is suggested: $C = [P_c, S_c, L_c, d_c, D_c, t_c, R_{UDID}]$ in which we have a new information containing a random constant unique identity for each robot that is generated when they start working, in this case, instead of having a hash table which stores the number of robots in each path, each robot can read crumbs and see how many unique identity are in this path. However, persistence of the crumbs still depend on the evaporation rate and the length of the trail. In order to solve this issue, since the path converges after a while, each robot knows an estimate of traveling time, so they can count the number of robots based on the crumbs where the creation timestamp is greater than the average traveling time.

Another way of making this approach distributed is having an agent at home which is incharge of maintaining the hash table and relays the information when a robot reaches home, based on which the robot decides the source to pursue. For this, we can either place a server like setup near home or arbitrarily have one of the robots stay at home to assume the role of conductor.

VI. EXPERIMENTS

VII. CONCLUSIONS AND FUTURE WORKS

In this paper, we outlined two approaches for improving LOST algorithm for better throughput in a multiple-source, single-home setting. We have provided experimental results showing improved performance compared to a variant of LOST (SO-LOST) both quantitavely and qualitatively.

In our future works we will implement a fully distributed version of our algorithm, potentially with real robots. Another extension can be implementing Maximum Sustainable Yield (MSY) [**?**] where the sources have a finite (and possibly different) resource growth rate. Previous works in multi-robot MSY foraging [**?**] and [**?**] rely on known source position. We can extend our current work to deal with simultaneous source finding and MSY in multiple-souce, single-home setting.

APPENDIX

Appendixes should appear before the acknowledgment.

ACKNOWLEDGMENT

The preferred spelling of the word acknowledgment in America is without an e after the g. Avoid the stilted expression, One of us (R. B. G.) thanks . . . Instead, try R. B. G. thanks. Put sponsor acknowledgments in the unnumbered footnote on the first page.

References are important to the reader; therefore, each citation must be complete and correct. If at all possible, references should be commonly available publications.

REFERENCES

[1] G. O. Young, Synthetic structure of industrial plastics (Book style with paper title and editor), in Plastics, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 1564.

[2] W.-K. Chen, Linear Networks and Systems (Book style). Belmont, CA: Wadsworth, 1993, pp. 123135.

[3] H. Poor, An Introduction to Signal Detection and Estimation. New York: Springer-Verlag, 1985, ch. 4.

[4] B. Smith, An approach to graphs of linear forms (Unpublished work style), unpublished.

[5] E. H. Miller, A note on reflector arrays (Periodical styleAccepted for publication), IEEE Trans. Antennas Propagat., to be publised.

[6] J. Wang, Fundamentals of erbium-doped fiber amplifiers arrays (Periodical styleSubmitted for publication), IEEE J. Quantum Electron., submitted for publication.

[7] C. J. Kaufman, Rocky Mountain Research Lab., Boulder, CO, private communication, May 1995.

[8] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, Electron spectroscopy studies on magneto-optical media and plastic substrate interfaces(Translation Journals style), IEEE Transl. J. Magn.Jpn., vol. 2, Aug. 1987, pp. 740741 [Dig. 9th Annu. Conf. Magnetics Japan, 1982, p. 301].

[9] M. Young, The Techincal Writers Handbook. Mill Valley, CA: University Science, 1989.

[10] J. U. Duncombe, Infrared navigationPart I: An assessment of feasibility (Periodical style), IEEE Trans. Electron Devices, vol. ED-11, pp. 3439, Jan. 1959.

[11] S. Chen, B. Mulgrew, and P. M. Grant, A clustering technique for digital communications channel equalization using radial basis function networks, IEEE Trans. Neural Networks, vol. 4, pp. 570578, July 1993.

[12] R. W. Lucky, Automatic equalization for digital communication, Bell Syst. Tech. J., vol. 44, no. 4, pp. 547588, Apr. 1965.

[13] S. P. Bingulac, On the compatibility of adaptive controllers (Published Conference Proceedings style), in Proc. 4th Annu. Allerton Conf. Circuits and Systems Theory, New York, 1994, pp. 816.

[14] G. R. Faulhaber, Design of service systems with priority reservation, in Conf. Rec. 1995 IEEE Int. Conf. Communications, pp. 38.

[15] W. D. Doyle, Magnetization reversal in films with biaxial anisotropy, in 1987 Proc. INTERMAG Conf., pp. 2.2-12.2-6.

[16] G. W. Juette and L. E. Zeffanella, Radio noise currents n short sections on bundle conductors (Presented Conference Paper style), presented at the IEEE Summer power Meeting, Dallas, TX, June 2227, 1990, Paper 90 SM 690-0 PWRS.

[17] J. G. Kreifeldt, An analysis of surface-detected EMG as an amplitude-modulated noise, presented at the 1989 Int. Conf. Medicine and Biological Engineering, Chicago, IL.

[18] J. Williams, Narrow-band analyzer (Thesis or Dissertation style), Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, 1993.

[19] N. Kawasaki, Parametric study of thermal and chemical nonequilibrium nozzle flow, M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.

[20] J. P. Wilkinson, Nonlinear resonant circuit devices (Patent style), U.S. Patent 3 624 12, July 16, 1990.