Blazing a trail: Insect-inspired resource transportation by a robot team

Richard T. Vaughan, Kasper Støy, Gaurav S. Sukhatme, and Maja J. Matarić

Robotics Research Laboratories, University of Southern California,
Los Angeles, CA 90089-0781

vaughan|kaspers|gaurav|maja@robotics.usc.edu

Abstract

We demonstrate a team of real robots that cooperate to robustly transport resource between two locations in an unknown environment. The robots use a trail laying and following algorithm inspired by the trail following of ants and the waggle dance of honey bees. Rather than directly marking their environment, the robots announce landmarks in their odometric localization space. The system tolerates significant odometric drift before it breaks down.

Keywords: robot team insect transport distributed

1 Introduction

In a recent paper we described a method for establishing a robot 'supply column' inspired by the trail-laying of ants and the waggle dance communication of bees. We demonstrated a group of simulated mobile robots that communicate by leaving landmarks in their shared localization space. By laying and following trails of such landmarks, the robots could robustly transport resources between two areas in an unknown environment [1].

The algorithm's robust performance in simulation, particularly with respect to localization errors, indicated that the method should be suitable for use in real robots. In this paper we present our first results with real robots. We briefly reiterate our motivation for tackling this task, then describe the trail laying algorithm and its implementation on a team of four Pioneer 2 DX robots. The results from an initial experiment are presented, showing that the method transferred successfully into the real world.

1.1 Ants, trails and localization

Ants form supply columns to relocate valuable items through complex, dynamic environments. Their remarkable effectiveness is due to a highly robust strategy of local interactions among a large number of autonomous agents.

The chemical trails formed by ants along their supply routes are robust with respect to changes in the environment and to the 'failure' of many individual ants [2,3]. These properties are attractive to agent designers in general and in particular to robot builders who can see immediate applications for robot supply columns in hazardous or tedious environments.

Ant-inspired solutions to various search problems have been demonstrated [4–8], and chemical trail laying and following has been demonstrated in robots [9,10]. Holland *et al* exploit stigmergy in a foraging task with real robots [11]. However it is often impractical and sometimes undesirable for robots to physically mark their environment, so in our method a group of robots deposits landmarks in a shared *localization space*.

We define localization space as any consistent spatial or topological representation of position. Such a space is *shared* if there is some (probably imperfect) correlation between the representations maintained by two or more individuals. A prime example is the Global Positioning System (GPS). Two systems equipped with GPS share a metric localization space in planetary coordinates. Similarly two robots that start out with known positions in the same coordinate system and maintain a position estimate via odometry share a localization space. In both examples each robot has only an *estimate* of its position in the true space, but the true space is common to both.

The robots in these experiments are localized by integrating their odometry to maintain a position and heading estimate. All robots are started from the same position so their coordinate systems are initially aligned. They share a localization space, but the correlation between the spaces will deteriorate over time as the error in each individual's localization estimate increases.

2 Task and approach

We examine the ant-like task of having multiple autonomous robots find and repeatedly traverse a path in an unknown environment between a known 'home' and a supply of resource at an initially unknown position.

Achieving this task reliably with robots would meet real current and future needs: a factory may require a supply of widgets manufactured at position A to be transferred by robot to position B; an army may require supplies (medicine, food, ammunition) to be transported over hazardous and uncertain terrain; a colony on Mars might want to to retrieve resources from a distant autonomous mining robot. The start location and the existence of one or more resource locations may be the only known features, and these may be a few meters or several kilometers apart. Establishing a reliable automated supply column robust with respect to loss of individual robots could be very valuable.

The experiments described in this paper demonstrate a simple algorithm for resource transportation by robot teams using mechanisms loosely analogous to the ant trail following and the honey bee 'waggle dance' [12]. Instead

Fig. 1. The Pioneer 2DX robots used in the experiments.

of directly modifying their physical environment like ants, or performing an information-transmitting dance like bees, our robots communicate localization space landmarks over a wireless network. This communication requires very little bandwidth and is well within the capabilities of current networks. Our communication strategies are informed by the work of [13] but are more complex because we are tackling a more realistic task in a more complex environment.

3 Robots

The experimental platform for these experiments was four identical ActivMedia Pioneer 2DX robots. These are fairly small (40x50cm), differential-steering and fitted with PC104+ Pentium computers running Linux. Each robot has front and rear sonar rings and wheel encoders as its basic sensors. For these experiments a SICK LMS scanning laser rangefinder was fitted to the front of each robot, providing excellent quality range readings over 180°. The SICK gives two samples per degree and an range accuracy of a few millimeters to most surfaces. BreezeNet 802.11 wireless Ethernet connected the robots and our workstations with a nominal bandwidth of 3Mbits/sec. The robots are shown in Figure 1, named Ant, Bee, Punky and Tanis.

4 Robot controller

4.1 Trail laying algorithm

Robots start from a home position and search for the resource. On reaching the resource, they receive a unit of resource and must return home with it, then return to fetch more resource repeatedly for the length of the trial. Each robot records its movements as a sequence of landmarks in localization space and shares the path with its team mates after each successful traverse. The algorithm consists of three decision processes running in parallel. A schematic is shown in Figure 2.

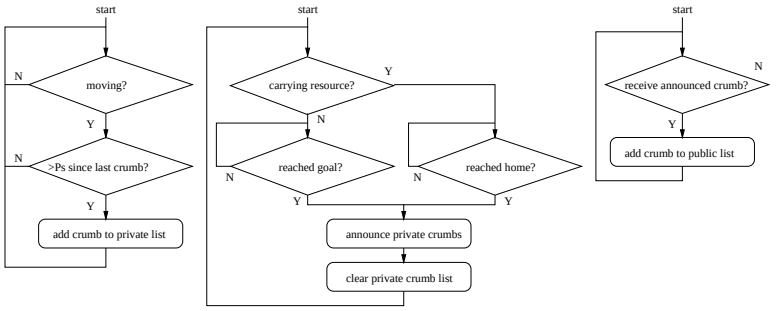

Fig. 2. Schematic of the trail laying algorithm. Three decision processes run in parallel to manipulate the private and public crumb lists.

The trail laying algorithm is independent of the method used to drive the robot's wheels; rather it can provide a hint of a 'good' heading to go in from the robot's current location. In this implementation the robots initially search the environment by random exploration: following walls and turning at random at intersections. This was chosen because it is simple, requires no *a priori* knowledge of the environment and will always find a path if one exists, given sufficient time. As it moves through the environment each robot records its current position and heading estimate at regular time intervals. We refer to this as 'crumb dropping'. As time goes by, a robot stores many such records on its initially empty *private crumb list*.

If a path to the resource exists, a robot will eventually reach it. Whenever a robot reaches the resource it announces the contents of its private crumb list on the broadcast channel of the shared network. All the robots (including the sender) receive the crumbs and add them to their initially empty *global crumb list*. The sender's private crumb list is cleared. Communicating the route only upon completing a successful trip corresponds loosely to the honey bee's waggle dance, as compared to the ant which leaves a trail as it goes along.

The first robot to reach the resource must have used the most time efficient path yet discovered and all robots now have a list of waypoints describing this path. At each timestep each robot examines its global crumb list and computes the average heading of all those crumbs that lie within a threshold radius of its current position estimate. If there are no such crumbs, the robot performs random exploration, otherwise it does wallfollowing and turns in the direction of the average heading at intersections. If a robot is carrying resource, i.e. it has reached the supply of resource, it will follow the opposite heading back to the home position. On reaching home a robot also broadcasts and clears its private list.

Thus a robot whose position estimate is similar to the position of a dropped crumb will tend to move in the same direction as the successful

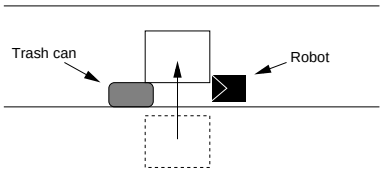

Fig. 3. Principle of the sliding box algorithm. The free-space box is initially placed to the left of the robot and is slid in the direction of the arrow until there are no obstacles detected inside the box.

robot that passed that way earlier. By following a trail of crumbs each robot can find the resource much faster than by random exploration.

4.2 Navigation

Navigation and obstacle avoidance is provided by two simple high-level behaviours: *Navigate* and *panic*. *Navigate* drives the robot around the environment following walls, turning down corridors and following crumb trails. Navigate is described in detail below. An emergency-stop mechanism runs continuously in parallel, monitoring the laser sensor; if the robot gets too close to any obstacle it will stop and switch to the *panic* behaviour.

Panic is designed to cope with situations that confound *navigate*. It 'unsticks' the robot from obstacles, dead ends and traffic jams by turning on the spot in a random direction until it detects free space and moves forward. After a few seconds the behaviour times-out and switches to the *navigate* behaviour.

The navigate behaviour is fairly sophisticated and contains several sub behaviours. It exploits the accuracy of the SICK laser scanner to make precise movements, turning tight corners and closely following walls.

At each time step the behaviour detects corridors to the left and right of the robot using the laser scanner. A corridor is identified if the extreme left and rightmost laser samples show a nearby obstacle followed by a large discontinuity and an opening wide enough to accommodate the robot. If the robot detects a corridor and it is not already turning, there is a 50% chance that the robot will decide to take the new corridor and make a sharp 90^o turn into the corridor. The corridor-seeking behaviour is overridden by a trail-following component if the trail points away from the opening.

If the robot is heading in approximately the opposite direction to the trail it makes a sharp 180^o right turn. If there is no reason to make sharp turns the behaviour defaults to wall following.

The wall following behaviour is implemented using a sliding box algorithm. A virtual box a little bigger than the robot is placed to the left in the laser scan (see Figure 3). The box is moved left to right in front of the robot until the laser scan shows no obstacles within the box. The robot moves forwards

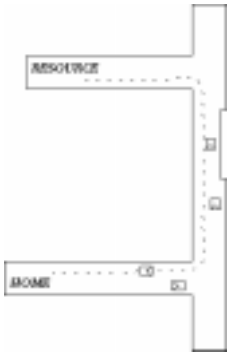

Fig. 4. Diagram of the experimental environment; part of our office building. The dashed line is an example crumb trail. Robots (small rectangles) are drawn to indicate scale.

at a constant speed, turning towards the center of this box . If there are no obstacles in the first box that is placed it is assumed that the robot is in a big open space and therefore moves forward.

The advantage of this approach compared with approaches that try to keep a fixed distance to the wall is that the size of the box can be tuned so that the robot keeps minimum distance to the wall but still has time to move away from the wall to avoid obstacles. If no open space is found the control system turns the robot in the opposite direction of the nearest obstacle.

This navigation controller is carefully designed to make the most of the limited space available in our environment. Navigate will keep control of the robot and follow the trail unless the emergency stop mechanism detects an obstacle critically close to the front of the robot. These behaviours make it possible for the robots to perform their task in small rooms and narrow corridors, passing within as little as 40cm of each other before interference occurs.

5 Experiment

The system was tested in our office building, with the resource separated from the home position by two corners and approximately 33m of corridors. To follow the optimal route to the resource, the robots must make two correct turning decisions based on the hint govern by the crumb trail. The layout is shown in Figure 4 and is similar to that used in our recent simulation experiments [14].

The robots are started one at a time from the same spot at the home position, with the same orientation. This position is taken as the the origin of their odometric localization space. The controller is run and the robot starts to explore the environment. When a robot comes within 1m of the the

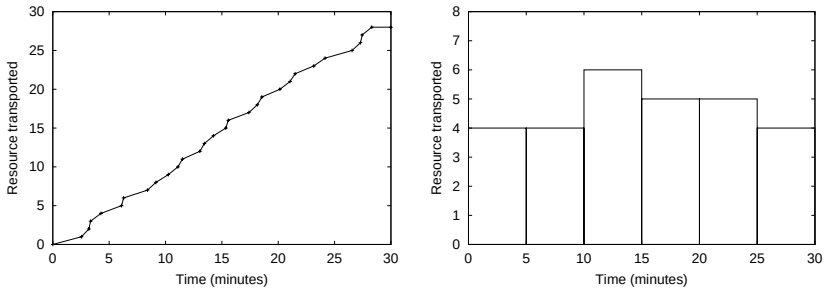

Fig. 5. Total units transported (left) and frequency of transportation (right).

home or resource position the experimenter notifies it by sending a message over the network. The trial runs for 30 minutes and the the robots are fully autonomous for the whole trial.

Every time a robot has completed a home-resource-home round trip (and therefore notionally transported a unit of resource) the time and the name of the robot is noted. Additionally the crumb trail broadcast by each robot is logged.

5.1 Results

Ant was the first robot to reach the resource, at 1.25 minutes. It broadcasts the first crumb trail to its colleagues. Tanis and Bee were already heading towards the resource, but Punky was heading in the wrong direction, back towards the home position. It was observed to immediately turn 180^o and follow the wall towards the resource. Ant turned around at the resource position and followed its trail back to the home position, correctly turning right at both corners. The other three robots also turned correctly and reached the resource by the optimal route.

All four robots continued to shuttle between home and resource for the next few minutes, until Punky and Bee came too close together as they negotiated the same corner in opposite directions. Their controllers went into *Panic* mode and they moved into open space, leaving the trail. Both then switched back into *Navigate* and quickly recovered the trail and followed it in the correct direction. Such near-collisions and recoveries occurred 5 times during the trial.

Until 20 minutes all robots made consistently correct turning decisions, but after 20 minutes they would occasionally make an incorrect choice, indicating that their localization estimates were diverging. After making one or two incorrect turns, the robots would apparently recover and follow the trail again. It may be that the robot wandered by chance into a region where the trail was better-formed than elsewhere, but we have insufficient data to analyse this at the time of writing.

Fig. 6. The trails announced by each robot over the 30 minutes trial, illustrating the the accumulating odometry errors and the large variation between robots.

Figure 5(left) shows the units of resource transported over time. The slope of the curve levels out at around 27 minutes; no more units were transported after this time. The robots seemed to suddenly lose any coherence in their movements. They appeared to turn at random and ended up jammed together in one spot until the end of the trial at 30 minutes.

The amount of resource transported by each robot was:

Tanis	Ant	Bee	Punky	Total
7	7	6	8	28

Since the time for a round-trip from home to resource is approximately 3 minutes the robot group performed three times better than a single robot performing under optimal conditions.

Figure 5(right) shows how the frequency of transported resource changes over time. It can be seen that the amount of resource transported per time unit decreases over time due to the increasing odometry error. The unbounded odometry error causes the system to break down completely at around 27 minutes.

In figure 6 the trails broadcasted by each robot is shown. It can be seen that the odometry of Ant and Bee is worse than the odometry of Tanis and Punky whose odometry only rotates slightly during the entire trial.

Despite the bad odometry of Ant and Bee these robots still perform well, indeed better than the authors expected. This is due to the robustness of the algorithm with respect to rotation in the environment; the average crumb heading does not have to point the robot in exactly the correct direction. When the robot detects an opening it turns into the opening if the average heading of the crumbs points to the side where the opening is detected. This

means that the robot will make turns correctly as long as the average crumb heading lies within 90^o of the direction of the opening. This provides a very high tolerance to localization error.

The robots also get a little help from the particular environment we used. The correct turn is the same at each of the two corners, so if the localization error is such that the two corners are confused, perhaps by a 90^o rotation, the 'correct' turn is taken. Future experiments will be design to avoid this advantage.

6 Conclusion and further work

We have shown that our insect-inspired trail laying algorithm is effective for a real robot team. Sharing landmarks in localization space instead of the real world allows us to achieve some of the benefits of stigmergic behaviour while avoiding the difficulties of laying and detecting physical trail-markers.

The trail-laying algorithm depends on the robots having a common localization space, but can tolerate a significant amount of error before it fails [1]. The system in this paper eventually breaks down due to the unbounded accumulation of odometry errors. The working life of the system could be extended by reducing the rate of drift, or if the localization error is kept within some relatively high bounds it will work indefinitely.

Methods for slowing odometric drift have been described by [15], and bounding drift by [16]. We aim to enhance our localization in the near future, likely using real-world landmarks to maintain the commonality of the robots' localization spaces.

Meanwhile we are investigating some immediate extensions of this work, such as moving home and resource positions, multiple sources of resource, and restricting the communication range of the robots.

The system's performance was degraded by near-collisions between robots. The phenomenon of interference is an important problem for multi-robot systems, particularly in constrained spaces such as home and office environments. In these experiments we deliberately worked only in the corridors, where robots could always pass each other. We are extending our controllers to cope with co-localization conflicts, such as two robots passing in opposite directions through a narrow doorway. Results of our initial simulation work have been submitted elsewhere [14] and a similar approach will be implemented on the real robots in the near future.

Acknowledgements

Thanks to Brian Gerkey for endless and uncomplaining help with the Pioneer robots and Barry B. Werger for discovering a disconnected serial cable inside a robot. This work is supported by DARPA grant DABT63-99-1-0015, NSF grant ANI-9979457 and DARPA contract DAAE07-98-C-L028.

References

1. Richard T. Vaughan, Kasper Stoy, Gaurav S. Sukhatme, and Maja J. Mataric. Whistling in the dark: cooperative trail following in uncertain localization space. In *Proc. Fourth Int. Conf. Autonmous Agents*, to appear 2000.
2. B. Holldopler and E.O. Wilson. *The Ants*. Springer-Verlag, 1990.
3. O.W. Richards. *The Social Insects*, page 113. Harper Torchbook, 1970.
4. M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, pages 26(1):29–41, 1996.
5. J. L. Deneubourg, S. Aron, S. Goss, J. Pasteels, and G. Duerinck. The dynamics of collective sorting: Robot-like ants and ant-like robots. In *Proc. 1st Int. Conf. Simulation of Adaptive Behaviour*, pages 356–363, 1990.
6. Ralph Beckers, Owen E. Holland, and Jean Louis Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. In *Proc. Artificial Life IV*. MIT Press, 1994.
7. Simon Goss, Jean Louis Deneubourg, R. Beckers, and J. Henrotte. Recipes for collective movement. In *Proc. European Conference on Artificial Life*, pages 400–410, 1993.
8. Jean L. Deneubourg, G. Theraulaz, and R. Beckers. Swarm-made architectures. In F. Varela and P. Bourgine, editors, *Proc. European Conference on Artificial Life*, pages 123–133. MIT Press, 1992.
9. Titus Sharpe and Barbara Webb. Simulated and situated models of chemical trail following in ants. In *Proc. 5th Int. Conf. Simulation of Adaptive Behavior*, pages 195–204, 1998.
10. R.A. Russell. Ant trails - an example for robots to follow? In *Proc. 1999 IEEE International Conference on Robotics and Automation*, pages 2698–2703, 1999.
11. C. Melhuish, O. Holland, and S. Hoddell. Collective sorting and segregation in robots with minimal sensing. In *Proc. 5th Int. Conf. Simulation of Adaptive Behaviour*, 1998.
12. K. von Frisch. *Bees. Their Vision, Chemical Senses, and Language*. Cornell University Press, Ithaca N.Y., 1950.
13. C. Melhuish, O. Holland, and S. Hoddell. Using chorusing for the formation of travelling groups of minimal agents. In *Proc. 5th Int. Conf. Intelligent Autonomous Systems*, 1998.
14. Richard T. Vaughan, Kasper Stoy, Gaurav S. Sukhatme, and Maja J. Mataric. Go ahead, make my day: Robot conflict resolution by aggressive competition. In *Proc. 6th Int. Conf. Simulation of Adaptive Behaviour*, 2000 (submitted).
15. S.I. Roumeliotis, G.S. Sukhatme, and G.A. Bekey. Smoother based 3-d attitude estimation for mobile robot localization. In *Proc. 1999 IEEE International Conference in Robotics and Automation*, May 1999.
16. Sebastian Thrun. Learning maps for indoor mobile robopt navigation. *Artificial Intelligence*, 1:21 to 71, 1999.