ARTICULATED ROBOTS MOTION PLANNING USING FORAGING ANT STRATEGY

Mohd Murtadha Mohamad

Faculty of Computer Science and Information Systems
University Teknologi Malaysia
81300 Skudai, Johor

Email: murtadha@utm.my,

Abstract: Many different approaches to tackle the problem of motion planning for articulated robots in an environment with obstacles based on random sampling have been proposed. One popular approach is called single-query bi-directional motion planning with a lazy collision checking probabilistic roadmap (SBL-PRM). However, the performance of this method is sub-optimal in terms of the number of configurations generated, length of path, amount of collision checking and computational time. To improve the performance, those aspects must be considered further as they are inter-related with each other. A novel modification of SBL-PRM that decreases the size of excessive configurations in the roadmap, by incrementally building a one-tree structure originating from the start configuration, is presented. This approach, the single-query unidirectional approach with lazy collision checking (SUL-PRM), has experimentally shown to be equal to the SBL-PRM. However, there still exists generated configurations that were excluded from the successful path. The generation of these unconsumed configurations corresponding to the tree structure has pointlessly utilized the computational resources and affected the planning time. Hence, a new method of configuration generation along with a novel searching style is devised. An alternative search approach using ant behaviour in a robotics application is applied. This paper proposes a novel search technique, the F-Ant algorithm, in order to find a reliable path between the initial configuration and the goal configuration of the articulated robot. This novel algorithm, taking two input configurations, explores the robot's free space by building up a unidirectional search beginning at the initial configuration. The planner samples the free configuration repetitively in the neighbourhood within the radius of the current configuration, and tests the edge for a collision-free path between the new sampled configurations, until it is connected to the goal configuration. Simulation and experimental comparisons of F-Ant and SBL-PRM have been conducted, showing the performance differences between these two techniques.

Keywords: Articulated robot, motion planning, foraging ant.

1. INTRODUCTION

Motion planning problems are related to the computational problems that have bounds on time and space, or, in other words, is the size of the problem or computational complexity. The computational complexity of a motion planning problem grows quickly with the size of description of the environments; for example, for a polyhedron robot in a polyhedron workspace, the size of the problem is the total number of triangles describing the surfaces of the robot and all obstacles. This shows that solving motion-planning problems clearly takes a formidable amount of time if a complete algorithm is used. This has led some studies to propose alternative heuristic planning methods. Certain of these methods are adequate to solve difficult problems, but also unintentionally fail to solve easier problems. As a matter of fact, most heuristic methods offer no performance guarantee.

1.2 Probabilistic Roadmap

Probabilistic RoadMap planners (PRM) are motion planners that can be categorised between fully complete and heuristic planners. The work in this paper focuses on improving the state-of-the-art in PRM planning, by introducing new techniques and a combination of techniques to reduce the distance travelled by a PRM planner.

Over the past fifteen years, the famous motion planning technique, PRM, has been studied by many different researchers [1][2][3][4][5][6] The technique has been shown to be very efficient, easy to implement, and applicable to many different types of motion planning problems. Therefore, it is the approach adapted in this paper for use in benchmarking against new algorithms.

The single query bi-directional motion planning with lazy collision-checking (SBL-PRM) [7] has reduced the amount of unnecessary collision checking and the number of unimportant configurations in the configuration space, C from the Lazy-PRM [8] in order to find the solution path. However, there are still excessive numbers of unused configurations in the roadmap after the solution is found. The greater the number of unusable configurations, the more unnecessary collision checking has been done to generate them. If the number of the configurations are becoming relatively large, it will be insignificant compared to the Lazy-PRM in terms of a single-query problem resolver. This is because of the larger size of roadmap, produced from the SBL-PRM planner, is almost equivalent to the precomputed roadmap of Lazy-PRM planner that it can be reused for other queries (multiple queries). The precomputed roadmap is large so that it can cover almost the entire free space.

1.3 Natural Behaviour Approach

An alternative method is proposed to overcome the excessive configuration problem in order to retain the planner as a single-query problem solver. Recently, the idea of "learning from nature" has quickly spread through many scientific researches: computer science (artificial intelligence), engineering, and robotics. The main goal is to exploit the impressive results achieved by insects when foraging for food. The studies on social insects such as ants and bees have contributed to many areas including network routing, optimisation, finding the shortest path and navigation.

One of the special manners of the insects like ants is the organisation of their social societies. Because of this organisation, ant colonies can accomplish complex tasks that in some cases far exceed the individual capabilities of single ant. Another brilliant capability exhibited by natural systems is the navigational skills of insects.

1.4 Paper Organisation

Section 2 describes how the ant behaviour contributes to the motion planning. Section 3 shows the motivation of pursuing this research. Section 4 describes the foraging ant algorithm and Section 5 shows the experiments and results.

2. ANT BEHAVIOURS

The specific insects that have gained interest in this study are the ants. In this section, two main skills of ants will be described: performing tasks as a colony, and performing tasks as an individual. The ant colony behaviour is exploited mainly in the optimisation problems. On the other hand, individual ant behaviour is employed to the
greatest possible advantage in searching.

2.1 Ant Colonies

An important research insight into ants' behaviour was that most communication amongst individuals, or between individuals and the environment, is based on the use of a chemical produced by the ants, pheromone. For some ant species such as *Lasius niger* [9] or the Argentine ant *Irdomyrmex humilis* [10], the trail pheromone is a specific type of pheromone that is used for marking paths on the ground, generally, paths from food sources to the nest.

By detecting the pheromone trails, foragers can follow the path to food discovered by other ants. This collective trail-laying and trail-following behaviour in which an ant is influenced by a chemical trail left by other ants is the motivation of ant colony optimisation (ACO) [11][12][13]. The ant can find the shortest path from nest to food source, based on the above features. This process has been shown by Deneubourg et al. [10] and Goss et al. [14] where a double bridge is used to connect a nest of Argentine ants *Irdomyrmex humilis* and a food source. They varied the ratio between the lengths of the two branches of the double bridge.

In the first experiment, the bridge had two equal length branches . At the beginning, ants were left free to move between the nest and the food source. Although in the initial phase random choices happened, in the end all the ants used the same branch. This is because when the experiment starts there is no pheromone on the two branches. Therefore, the ants do not have a preference and they selected at random any of the branches. Because of random variations, a few more ants select one branch over others. Since ants lay pheromone while walking, larger amounts of pheromone are deposited on a branch passed by a larger number of ants. This larger amount of pheromone stimulates more ants to choose that branch again, until the ants concentrate on one single path. This is a form of *positive feedback*: the process that reinforces itself in a way that causes very rapid convergence. The convergence of the ants' paths to one branch represents the collective behaviour by the local interactions among the individuals of the colony. Some other types of ants, for example *Lasius niger* scout ants [9] laid pheromone only if they returned to the colony after finding the food source. This will allow them to recruit worker ants to trace the trails left by the scouts and retrieve the food to their nest.

Another experiment was performed on the double bridge test in which the length ratio of the two branches was set to two [14]. This means that one branch is twice as long as the other branch. After some time all the ants leave the nest to explore and arrive at a decision point where they have to choose one of the two branches. They choose randomly as the branches initially appear identical to the ants. The ants choosing the shorter branch are the first to reach the food and start returning to the nest. Here again they must make a decision between the short and the longer branch. The higher-level pheromone on the short branch will have greater influence on their decision. Hence, pheromones start to accumulate faster on the short branch that will be used by all ants because of the positive feedback process described above.

From the above analysis, it is found that the behaviour of the ant colony is a kind of group behaviour incorporated with the aid of the self-organising behaviour of the single individual. The importance feature of ACO is the positive feedback process shown in the group of ant colonies. With the help of positive feedback, the whole system can evolve gradually.

2.2 Individual Ant

The issue of how an insect finds its way between the nest and distant feeding sites has been investigated extensively in a species of social insects, the desert ant (genus *Cataglyphis*) [15][16][17][18]. Despite their diminutive brains, many insects accomplish impressive navigation tasks. *Cataglyphis*, for example, is able to explore its desert habitat for hundreds of meters while foraging and going back to its nest precisely and on a straight line. *Cataglyphis* cannot use pheromones to retrace its trail in order to return to its nest, since the pheromones evaporate in a few seconds because of the high ground temperatures. Studies have revealed many details about the behavioural range of skills and the underlying mechanisms that *Cataglyphis* employs when homing. So far, the researchers have found that the *Cataglyphis* exercise three main strategies: direction-following and path integration, visual piloting, and systematic search [17][19].

Path integration is based on compass information gained from the polarization pattern of the sky. It is assumed that the search space of the ant is a 2-D space. The integration of angles steered and distances travelled is based on approximate computational rules. The determination of distance travelled is contributed to by the sense of vision. The distance it has covered is assessed by integrating the self-induced retinal image speed over the time. To reduce the time needed to locate the goal, *Cataglyphis* employs a systematic search strategy [20] rather than a random one and, in addition, uses landmark information whenever available.

2.3 Ant-inspired Robotics

So far, the ant has shown two main features for its survival either as one colony or as an individual. These capabilities have inspired many researchers to manipulate the ant behaviours into many fields of studies, including routing [21][12], assignment [22], scheduling [23] and especially in robotics: mobile robots [24][18][25], crawling robot [26], flying robot [27], and mobile robot simulation [28][29][30].

A very close study of integrating ant behaviour with robotics is in the work by Russell [26] that investigates transferring the pheromone trail tracking capabilities of ants to robotics systems. A robotic ant has been built and equipped with a pair of odour sensing antennae. The trail following algorithm is developed following the behaviour of the *Lasius fuliginosus* ant. Another related work is by Sugawara and colleagues [25] which is the Virtual Dynamic Environment for Autonomous Robot (V-DEAR) for real robot experimentation. In this system, pheromones are replaced with a graphic projected on the ground. Mobile robots decide their actions following the colour of the projected computer graphic (CG). The purpose of their experiments is to examine the performance of the robots to work in a group through the foraging tasks.

Other practical experimentation, involving integration of the ant behaviour with mobile robots, is the area coverage problem. Gabriely and Rimon proposed an Ant-like spanning tree covering algorithm on a mobile robot-covering problem [24]. The mobile robot follows a spanning tree of the graph induced by the cells, while covering every point only once. The mobile robot is equipped with position and orientation sensors to recognize locally the 2-D size cells of the working area. The robots can also leave markers in the sub cells it covers for identification of covered or uncovered cells, just like the behaviour of ants leaving pheromone trails.

Instead of the pheromone manipulation in robots in the above examples, a specific algorithm regarding ant colonies has been implemented which is ACO. The first problem encountered by ACO was the routing problem, specifically the travelling salesman problem (TSP) [11][31][12]. The TSP is the problem of a salesman who, starting from his hometown, wants to find a shortest tour that takes him through a given number of customer cities and then backs home, visiting each customer city only once. ACO algorithms have shown their efficiency in solving this problem. One of the practical experiments that involves ACO-TSP with a real robot is reported in the work by Agarwal et al. [27]. The purpose of their study was to find the shortest complete coverage flight path that satisfies the non-holonomic constraints of an unmanned reconnaissance aerial vehicle (URAV). The URAV mounted sensor has a small square footprint area for assessment of damage to physical assets at an airbase upon hostile action by an adversary.

Instead of practical implementations on a real robot, the ACO has also gained interest in robot simulation, such as temporal learning skills incorporated with ACO for robot navigation problems [30] and the implementation of the ACO-TSP to search for the shortest path between two locations [29]. Both approaches have discretised their 2-D workspace for specifying the cells of free space and the obstacles. Another work that involves ant behaviour in computer simulation is the Intensified ACO for continuous function optimisation problems [28]. The approach is applicable for mobile robots in an environment that changes rapidly.

For the non-pheromone ant navigation, Lambrinos and colleagues [18] developed mechanisms for path integration and visual piloting that are employed on the mobile robot *Sahabot* 2. They implemented a biological model of visual landmark navigation using a panoramic visual system. The robot follows the behaviour of desert ant *Cataglyphis*.

The next section gives the motivations upon generating the new algorithm, F-Ant, based on the ACO and its compatibility.

3. MOTIVATIONS

It has been shown that the ant behaviours, either as a colony or as an individual, have contributed to the field of robotics including cooperative robots, and navigation. The ACO has been implemented in some of the problems of area covering and searching for the shortest path which are the interest of this study. However, a few problems have arisen regarding ACO with the robot motion planning problem including the following:

- None of the problems described above is related to the articulated robot.

  In order to manipulate the ant capability for the articulated robot type, the problem should be reconstructed. Even when the robot is represented in C, the approach in ACO may be applied to the maximum of the 2-degrees of freedom articulated robot. The implementation will be rather trivial as less practical applications can be contributed for the 2-degrees of freedom articulated robot. Most of the industrial robots have at least 6-degrees of freedom, such as PUMA 560 and IRB 2400 robots.

- Additional constraints should be added to the ACO which include avoiding obstacles and the other robots (if they exist in the same environment for multiple robots) instead of distance constraints.

  ACO specifically stated that the inter-cities distances are computed on the fly distance between them [13]. This is clearly shown when the paths are straight. In the articulated robot case, the straight path can be defined in two terms; in cartesian space, the tip of the robot (mostly referred to as the wrist of the arm) moves in a straight line from one position to another position, while the other joints (the joints between shoulder and wrist) have to to change accordingly through the inverse kinematics; in configuration space, all the joints move from one configuration to another configuration at the designated angle at once (each joint may not necessarily move in the same direction). These, however, are applicable to the robot in an obstacle-free environment. If there are obstacles between two positions or configurations, the obstacles must be avoided along the path. This will result in the robot having to move to intermediate positions or configurations before reaching the goal configuration.

- ACO needs pre-computations before performing the actual tasks, such as finding the shortest tour in TSP.

  The ACO algorithms need four matrices of dimension of $n \times n$ (n is the number of cities) for representing: the intercity distance matrix, pheromone trails matrix, heuristic information matrix, and combination of pheromone and heuristic information matrix. Furthermore, in addition to intercity distance matrix, the nearest-neighbour lists matrix is needed with a size of $n \times n$. For each of the ants, two arrays are needed of size $n + 1$ and n to store, respectively, the tour and the cities visited, as well as the tour's length. The ACO may use m number of ants up to n. This is practical for the problems that deal with a finite set of cities which are the set of the coordinates (x,y) in 2-D space. Pre-computations are necessary for TSP problems as they have more than one goal (usually it takes in order of ten to a hundred cities). Moreover, as mentioned in the preceding problem, the path between cities is free of obstacles and can be accessed in a straight line. To apply the same strategy to articulated robot problems, the pre-computations in ACO will have the same characteristic as PRM, in which the motion planner pre-computes the roadmap before the queries were answered.

- Furthermore, for a mobile robot type, whether it is a real robot implementation or simulations, and either using ACO or other techniques to approach the problem, the representation of the ants is in the 2-D search space which is concerned with the positioning of the robot in a x, y coordinate system and with configurations of x, y, for a non-holonomic robot (mobile robot, e.g. car-like robot). The workspace is discretised beforehand so that the free space area and obstacles can be identified. This means that the algorithms must be specific to a certain condition which is having advance information about the environment before starting the navigation. Even though there is a crawling robot and a flying robot (e.g. URAV), the search space for either tracking pheromones or area covering is in a 2-D zone.

  For articulated robot problems, the robot and the environment cannot be simply represented in 2-D space. For some cases, if the robot has 2 rotating joints and both joint axes are parallel, it can be shown to be similar to a 2-D representation. The dimensions of the robot depend on the number of joints of the robot (n-joints robot have n dimensions). However, when the robot has more than 2 joints or the joint axes are not parallel, the representation cannot be shown in a 2-D representation. This will change the approach of tackling the problem if using the ACO.

For all the reasons above, it can be concluded that for articulated robot problems in a 3-D environment, the ACO is not practically suitable. The planner is supposed to generate a collision-free path from start to goal configuration. Nevertheless, one aspect of the general ACO algorithm that is close to the approach of articulated robot motion planning is improving the ants' tour. Even though this is only part is of the whole solution of TSP (decision to choose which point to visit from the current point in order to complete touring all the points in the problem environment), it is a crucial element of the robot motion planning problem.

However, some characteristics of the ant are applicable in solving the problem in planning the motion of the articulated robot. This will be described in the following section.

4. FORAGING ANT ALGORITHM

The strategy of the implemented ant is a combination of the behaviours of the Argentine ant forager which lays a trail while searching for food [10], and the desert ant which using a direction-following based on external reference sources for homing trajectories [20]. In this approach, the foraging ant already knows the location of the food source (external reference), which the ant has a reference point for its objective. However, this information is for the purpose of checking whether the ant can move, from its current location, straight to the food without any intermediate stopping. The ant has a vision for a specific range, ρ, to see the spot where it can stand before it makes any moves.

First, the ant is released from the nest and observes whether it can make it straight away from the nest to the food source. If it is not possible because of obstacles obstructing the path, or the vision is clear but the path cannot be constructed because it needs intermediate points in between, the ant will identify a random spot (sampling the space around) to make a next step. The ant will see to the furthest range of and make sure the path along the current spot to the candidate spot is free from obstacles and can be made without any intermediate stops (collision-free local path). The path between the current spot to the new spot is the local path for the ant. If it happens that the ant vision is on the obstacles, the ant will sample another random candidate spot and reduces its vision range by a factor of two. The sampling is repeated until the candidate spot is obstacle-free and is a collision-free local path, or the vision reduction has reached the limit (a number of reduction times will be specified). The process is repeated again until the ant can find the obstacle-free and collision-free local path. When the ant has found one, the ant will leave a trail on its current position and record it in its memory before moving to the new selected spot. At the new spot, the ant will look for the food location and check whether it can reach there, similar to the process when it was first released from the nest. If it still does not succeed, the ant will repeat the foraging process above until it reaches the objective or until the ant has reached its number of steps limit. The cumulative local paths between the nest and the food source will form a final path for the ant.

4.1 Overall Algorithm

The problem is formulated as follows. Given an n-dimensional configuration space C, let F be the subset of free configurations of C. Let q_{init} in F and let q_{goal} be a target, contained in F. Let Q_{goal} be the set of configurations that lies in F and can be connected straight to q_{goal}. The task is to find a path connecting q_{init} with Q_{goal}, that is, a continuous function f: $[0,1]\rightarrow F$ such that

$f(0) = q_{init}$ and $f(1)=Q_{goal}$. Assume that the values for each robot's degrees of freedom has been normalised to lie in the interval [0,1]. The planner is given a distance threshold parameter, ρ. The L_∞ vector norm metric is used over the robot configuration C which is defined as the maximum of the absolute value of the overall components a vector. Let $q_a= (q_{a1}, q_{a2}, \dots , q_{an})$ and $q_b= (q_{b1}, q_{b2}, \dots , q_{bn})$. The absolute distance value between q_a and q_b is $|d(q_a, q_b)| = |q_{a1}- q_{b1}|, |q_{a2}- q_{b2}|, \dots , |q_{an}- q_{bn}|$. So L_∞ distance is max $(|d(q_a, q_b)|)$. In other words, the distance between the two configurations q_a and q_b is denoted by $d_\infty : q_a , q_b \in C \rightarrow d_\infty (q_a, q_b) \in R$. The local path (q_a, q_b) is the segment connecting q_a and q_b. Here, ρ is typically set between 0.15 and 0.95.

The Foraging Ant (F-Ant) algorithm is as follows:

```
Algorithm 1 F-Ant($q_{init}$, $q_{goal}$)
1: k←0
2: if SegmentCheck($q_{init}$, $q_{goal}$)= TRUE then
3:       return SUCCESS
4: end if
5: $\tau.top \leftarrow q_{init}$ and $\tau.bottom \leftarrow q_{goal}$
6: while $k <= maxstep$ do
7:       while SegmentCheck($q_k$, $q_c$) FALSE, discard $q_c$ and do
8:             GenerateNode $q_c$
9:       end while
10:      $k = k + 1$
11:      $q_k \leftarrow q_c$
12:      $\tau \leftarrow q_k$
13:      if $q_k \in Q_{goal}$ then
14:            SUCCESS
15:      end if
16: end while
```

The step, k, is reset to zero at line 1. On lines 2 to 4, the connection is checked for collision between q_{init} and q_{goal}. If the test returns not TRUE, the search begins. A list of vectors, τ, is used for accumulating all the successful configurations in the form of q. At the beginning (line 5), q_{init} is inserted into τ at the top of the list and the q_{goal} at the bottom. For any new configuration, q_{new}, it will be inserted between q_{init} and q_{goal}. The loop in lines goal 7 to 9 is for generating the candidate node, q_c. k is increased when the q_c is found and the segment connecting q_c and q_k lies entirely in F. Then q_c is assigned as updated q_k and added to the configurations list τ in lines 11 and 12. q_c is tested whether it lies in Q_{goal}, and the algorithm will return SUCCESS if $q_k \in Q_{goal}$. The loop of line 6 to 16 is repeated until the path is found or when it reaches the maximum step, *maxstep*.

5. EXPERIMENTAL RESULT

F-Ant is written in ANSI C++ using the Standard Template Library (STL) in Microsoft Visual Basic 6.0. It was compiled on a PC running Windows XP. All experimental results reported in this paper were obtained on a Pentium- 4 PC with a 2.60-GHz processor and 512-Mb of memory. The following sub-sections describe the techniques that were designed as in Algorithm 1 to implement some of the key steps of the planner.

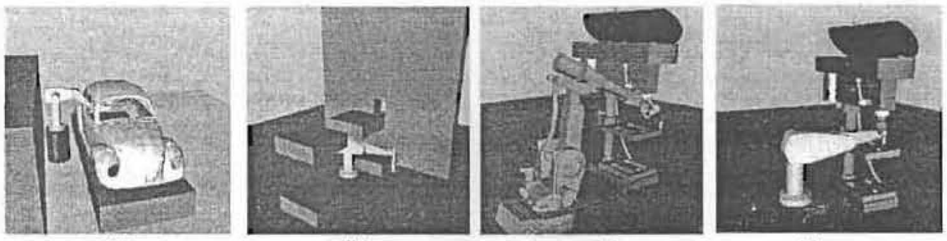


Figure 1 Motion planning environment.

The scenarios in **Figure 1** are those which have been used to test the F-Ant planner. In each scenario, let n_r and n_o denote the numbers of triangles in the geometric models of the robot and the obstacles respectively. One scenario is corresponding to a mobile articulated robot, whereas the rest is the floor attached articulated robot for each scenario.

- The mobile robot in **Figure 1**(a) is a 6 degrees of freedom PUMA560 robot arm mounted on a 3 degrees of freedom platform. The obstacles are a car body that contains both thin and narrow components and a long wall beside the car with a space that allows the robot to move in between. The robot has much maneuvering space. In this scenario, $n_r = 868$ and $n_o = 4, 040$.
- The scenario of Figure 1(b) consists of a floor-attached PUMA560 robot arm equipped with a torch. The obstacles are the high wall with a window, two floating cuboids and two other cuboids on the floor. Here, $n_r = 868$ and $n_o = 96$.
- The robot in scenario of Figure 1(c) is another 6 degrees of freedom fixed to the floor robot arm, IRB2400, equipped with an arc welding gun. The obstacle is a drilling machine. In this case, $n_r = 3,791$ and $n_o = 34,171$.

Figure 1(d) is a fixed PUMA560 robot arm, equipped with a torch. The obstacle is a drilling machine. $n_r = 868$ and $n_o = 34,171$.

5.1 Impact On The Configurations Generated

Table 1 shows results of number of configurations generated. The first column shows the scenes and the second column shows the value of. The third and fourth columns show the percentage of successful planning for the F-Ant and the SBL-PRM respectively. The fifth, sixth, and seventh columns show the total of configurations generated for F-Ant (in path) and the SBL-RM (in path and in roadmap, RM) respectively. Finally, the eighth, ninth, and tenth columns show the average of configuration generated in all successful paths for F-Ant (in path) and the SBL-RM (in path and in roadmap, RM) respectively.

Table 1: Statistics For The Number Of Configurations Generated For The Different Scenarios Using F-Ant And SBL-PRM

				Total			Average		
				F-Ant	SBL		F-Ant	SBL	
		success rate		in	in	in	in	in	in
Scene	ρ	F-Ant	SBL	path	path	RM	path	path	RM
1(a)	0.15	90.0	100.0	390,167	43,914	798,940	434	44	799
	0.55	100.0	100.0	165,340	28,730	1,070,971	165	29	1,071
	0.95	100.0	100.0	131,974	33,856	3,658,660	132	34	3,659
1(b)	0.15	99.5	100.0	243,357	21,667	150,920	245	22	151
	0.55	100.0	100.0	50,020	26,535	1,090,316	50	27	1,090
	0.95	100.0	100.0	37,040	30,416	2,798,433	37	30	2,798
1(c)	0.15	100.0	100.0	78,065	18,442	875,479	78	18	875
	0.55	100.0	99.8	12,547	18,264	6,370,950	13	18	6,384
	0.95	100.0	97.8	11,030	20,581	7,533,279	11	21	7,703
1(d)	0.15	100.0	100.0	43,648	18,490	623,299	44	18	623
	0.55	100.0	98.7	8,574	20,438	7,761,434	9	21	7,864
	0.95	100.0	92.0	7,383	22,454	10,606,333	7	24	11,529

Table 2: Percentage Of Configurations That Are Applied To The Actual Path From The Total Of Configuration Generated In Roadmap, RM, By Using The Algorithm SBL-PRM

		ρ	
Scenes	0.15	0.55	0.95
1(a)	5.50	2.68	0.93
1(b)	14.36	2.43	1.09
1(c)	2.11	0.29	0.27
1(d)	2.97	0.26	0.21

The overall results showed that ten out of twelve scenes with a different values of ρ gave 100% success with the F-Ant algorithm while the SBL-PRM just managed to get eight scenes successfully. The result has shown that the F-Ant managed to reduce a relatively large amount of configurations that it needs to apply in the actual path. For the value of $\rho = 0.15$, the number of configurations produced from the F-Ant algorithm is greater than the other values of ρ of the same algorithm. However, when ρ is increased to 0.55 and 0.95, the number of configurations is also reduced to about five times smaller. The F-Ant algorithm uses all of the configurations it has produced in the actual path. On the contrary, the SBL-PRM uses between 0.21% to 14.36% configurations only in the actual path (**Table 2**). This has shown that more than 80% of the generated configurations in the roadmap of SBL-PRM are useless. The result of F-Ant in scenario 1(a) is significant compared with the other scenarios because the number of degrees of freedom used was 12 degrees of freedom: nonetheless the others were 6 degrees of freedom.

5.2 Impact On The Length of Path

Table 3 gives the length of the path for each successful planning. The first and second columns show the scenes and the value of respectively. The third and the fourth show the total length produced from planning for F-Ant and SBL-PRM correspondingly. The fifth column shows the average length in a path produced by F-Ant and the sixth column is for the SBL-PRM. Lastly, the seventh column is the standard deviation length (Std. Dev.) of F-Ant and SBL-PRM in the eighth column.

Table 3: Statistics For The Length Of Path For The Different Values Of Using The F-Ant And SBL-PRM

		Total		Average		Std. Dev.	
Scene	ρ	F-Ant	SBL	F-Ant	SBL	F-Ant	SBL
1(a)	0.15	73,385.47	10,285.00	81.63	10.29	44.33	2.63
	0.55	89,121.72	15,500.33	89.12	15.50	78.63	5.46
	0.95	109,965.39	23,972.59	109.97	23.97	107.47	9.00
1(b)	0.15	37,937.90	4,464.69	38.21	4.46	28.41	1.37
	0.55	21,785.72	11,500.86	21.79	11.50	21.40	4.51
	0.95	23,210.14	15,588.34	23.21	15.59	20.54	6.98
1(c)	0.15	15,302.36	4,318.98	15.30	4.32	16.08	1.64
	0.55	7,040.44	10,625.08	7.04	10.65	5.93	4.46
	0.95	8,152.23	14,469.28	8.15	14.79	7.28	6.69
1(d)	0.15	6,928.06	3,856.78	6.93	3.86	8.88	1.59
	0.55	3,818.25	10,903.96	3.82	11.05	3.05	5.60
	0.95	4,041.88	14,289.79	4.04	15.53	2.89	8.29

The length of the successful paths produced by the F-Ant algorithm are rather large in scenarios 1(b), 1(c), and 1(d) when the value ρ of is 0.15. However, in scenario 1(a), the length is almost smaller at the same value of ρ. It can be assumed that the larger degrees of freedom of a robot (12 degrees of freedom in this case), the less path distance the planner will achieve. This is because of more manoeuvring capability by the robot in scenario 1(a), and it has more chance to sample a free space at a closer neigbourhood (with small ρ), while in the 6 degrees of freedom robot scenes, the middle value of ρ gives the shortest length of the successful path on average for F-Ant.

The SBL-PRM produced the shortest length in the successful path for the smallest ρ value in all scenarios. Thus, the SBL-PRM gave a greatest length when ρ is the biggest. It can be assumed that the length of path is proportional to ρ for SBL-PRM implemented scenarios.

The F-Ant managed to produce a shorter length for scenarios 1(c) and 1(d) when ρ are 0.55 and 0.95.

Table 4 shows the total collision checking performed by the F-Ant while the next column to it is the total collision checking by the SBL-PRM. The average of collision checking performed by F-Ant and SBL-PRM is shown in the fifth column and the sixth columns respectively.

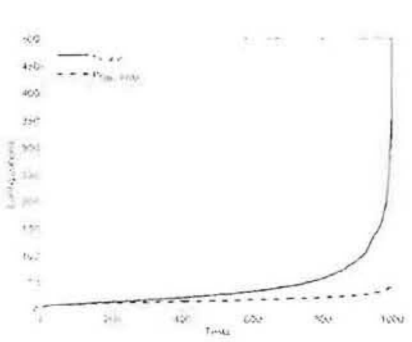
Table 4: Statistics For The Amount Of Collision Checking For The Different Values Of Using The F-Ant And SBL-PRM

		Total		Average	
Scene	ρ	F-Ant	SBL	F-Ant	SBL
1(a)	0.15	1,675,050	3,260,830	1,863	3,261
	0.55	5,996,495	15,183,182	5,996	15,183
	0.95	16,764,695	59,859,421	16,765	59,859
1(b)	0.15	1,018,453	556,120	1,024	556
	0.55	1,548,703	5,057,929	1,549	5,058
	0.95	2,854,626	13,088,914	2,855	13,089
1(c)	0.15	707,590	1,186,598	708	1,187
	0.55	5,369,341	7,111,191	5,369	7,125
	0.95	4,730,972	8,257,357	4,731	8,443
1(d)	0.15	740,613	1,264,930	741	1,265
	0.55	5,978,875	9,359,844	5,979	9,483
	0.95	7,762,945	12,414,394	7,763	13,494

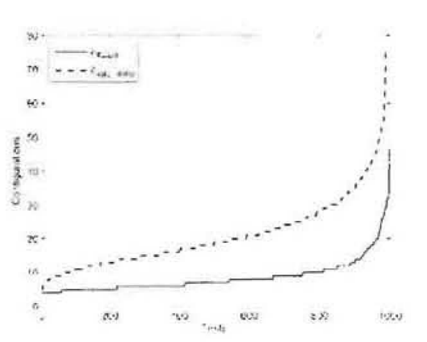
The overall result has shown that the amount of collision checking is proportional to the value of ρ. For all scenarios and all values of ρ, the F-Ant algorithm has performed less collision checking than the SBL-PRM either in total or on average. The main reason for this situation is that the SBL-PRM has restricted its collision checking upon generating its milestones (configurations) in the roadmap: for example up to 30,000 tests were performed in these experiments to generate a collision-free configuration. On the contrary, F-Ant allows an infinite number of collision-checking (exhaustive) in order to produce a collision-free configuration, but it restricted the number of steps to build up a successful path; in this article for example 1,000 steps is the maximum number of configurations allowed for each successful path. The other reason is that the F-Ant tests only the collision for the candidate configuration to be included in the path, while SBL-PRM tests for every single milestone, whether it is to be included in the path or not.

The results shown on **Figure 2** and **Figure 3** are referring to the experiment on **Figure 1**(d). **Figure 2** has shown that at the value of $\rho = 0.15$, the SBL-PRM generates less number of configurations in the successful path. However, when the value of ρ has increased to 0.55 and above, the F-Ant produced lesser number of configurations than SBL-PRM. While in **Figure 1** (d), computational time of F-Ant is lesser than SBL-PRM for every tested value of ρ.

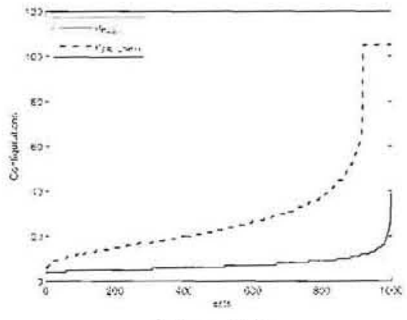
In terms of selecting of which algorithm should be applicable in practice corresponding to ρ value and the number of configurations, one possible solution is to consider the robot's manoeuvrability. If the robot is mobile manipulator, SBL-PRM with lower value of ρ is the better selection. However, if the robot is attached on the floor, F-Ant is the promising selection with setting the high value of ρ. One should also consider the time calculation factor because, even though F-Ant produced more configurations in greater manoeuvrability scenario, the calculation times were much quicker when the ρ are bigger.



(a) ρ =0.15

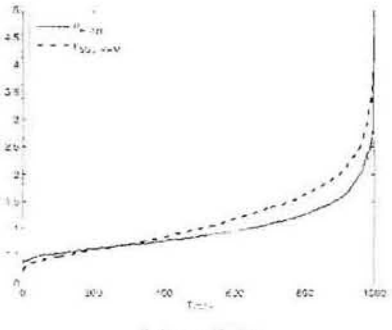


(b) ρ =0.55

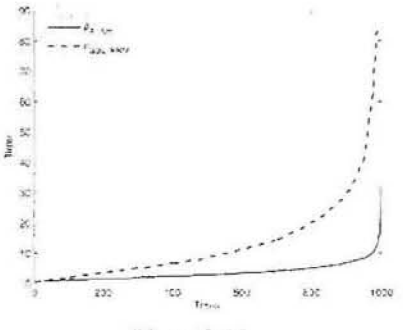


(c) ρ =0.95

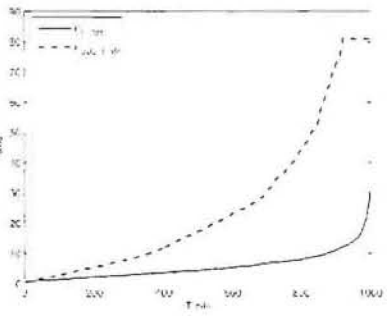
Figure 2 Comparisons of number of configurations in successful path between F-Ant and SBL-PRM



(a) ρ =0.15



(b) ρ =0.55



(c) ρ =0.95

Figure 3 Comparisons of computation time between F-Ant and SBL-PRM

6.0 CONCLUSION

A foraging ant motion planner has been introduced. A simple but effective technique has been devised by letting the ant act as a planner where it moves randomly and always checks the free path to the goal region. It can be concluded that the longer the free-obstacle range that the planner could sample towards the goal from its current configuration, the fewer number of steps it takes to build the path. And also F-Ant contributed to a faster computation to find the solution path.

REFERENCES

[1] N.M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In IEEE International Conference on Robotics and Automation, pages 113-120, 1996.

[2] Jérô Barraquand, Lydia Kavraki, Jean-Claude Latombe, Rajeev Motwani, Tsai-Yen Li, and Prabhakar Raghavan. A random sampling scheme for path planning. International Journal of Robotics Research, 16(6):759–774, 1997.

[3] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation , 12(4):566–580, 1996.

[4] L.E. Kavraki, M.N. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. IEEE Transactions on Robotics and Automation , 14(1):166–171, 1998.

[5] Lydia E. Kavraki. Random networks in configuration space for fast path planning. PhD thesis, Stanford, CA, USA, 1995.

[6] Lydia E. Kavraki and Jean-Claude Latombe. Randomized preprocessing of configuration space for fast path planning. In Proc. IEEE International Conference on Robotics and Automation , pages 2138–2145, 1994.

[7] G. Sanchez and J. Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In International Symposium on Robotics Research, pages 403–418, 2001.

[8] R. Bohlin and L. E. Kavraki. Path planning using Lazy PRM. In Proc. IEEE International Conference on Robotics and Automation, volume 1, pages 521–528, 2000.

[9] St'ephane Portha, Jean-Louis Deneubourg, and Claire Detrain. How food type and brood in uence foraging decisions of Lasius niger scouts. Animal Behaviour, 68(1):115–122, 2004.

[10] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. The self-organizing exploratory pattern of the Argentine ant. Journal of Insect Behavior, 3(2):159–168, 1990.

[11] Marco Dorigo, Gianni Di Caro, and Luca M. Gambardella. Ant algorithms for discrete optimization. Artificial Life , 5(2):137–172, 1999.

[12] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. The Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics , 26(1):29–41, 1996.

[13] Marco Dorigo and Thomas Stü¨utzle. Ant Colony Optimization . The MIT Press, Cambridge, Massachusetts, 2004.

[14] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. Self-organized shortcuts in the Argentine ant. Naturwissenschaften , 76(12):579–581, 1989.

[15] M. Collett, T. S. Collett, S. Chameron, and R. Wehner. Do familiar landmarks reset the global path integration system of desert ants? Journal of Experimental Biology , 206:877–882, 2003.

[16] T.S. Collett. How are studies of insect navigation useful to roboticists? In Proc. IEE Self-Learning Robots II: Bio-robotics (Digest No. 1998/248) , page 8/1, 1998.

[17] Matthias O. Franz and Hanspeter A. Mallot. Biomimetic robot navigation. Robotics and Autonomous Systems , 30(1-2):133–153, 2000.

[18] Dimitrios Lambrinos, Ralf Möller, Thomas Labhart, Rolf Pfeifer, and Rüdiger Wehner. A mobile robot employing insect strategies for navigation. Robotics and Autonomous Systems , 30(1-2):39–64, 2000.

[19] Rüdiger Wehner, Barbara Michel, and Per Atonsen. Visual navigation in insects: coupling of egocentric and geocentric information. Journal of Experimental Biology, 199(1):129–140, 1996.

[20] Rüdiger Wehner and Mandyam V. Srinivasan. Searching behaviour of desert ants, genus Cataglyphis (Formicidae, Hymenoptera). Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology , 142(3):315–338, 1981.

[21] B. Bullnheimer, R.F. Hartl, and C Strauss. An improved ant system algorithm for the vehicle routing problem. Annals of Operations Research , 89(1):319–328, 1999.

[22] V. Maniezzo and A. Colorni. The ant system applied to the quadratic assignment problem. IEEE Transactions on Knowledge and Data Engineering , 11(5):769–778, 1999.

[23] A. Bauer, b. Bullnheimer, R.F. Hartl, and C. Strauss. An ant colony optimization approach for the single machine total tardiness problem. In Proceedings of the Congress on Evolutionary Computation , volume 2, pages 1445–1550, 1999.

[24] Y. Gabriely and E. Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. In Proc. IEEE International Conference on Robotics and Automation, volume 2, pages 1927–1933, 2001.

[25] K. Sugawara, T. Kazama, and Watanabe. Foraging behavior of interacting robots with virtual pheromone. In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems , volume 3, pages 3074–3079, 2004.

[26] R. A. Russell. Ant trails - an example for robots to follow? In Proc. IEEE International Conference on Robotics and Automation, volume 4, pages 2698–2703, 1999.

[27] A. Agarwal, Meng-Hiot Lim, Meng-Joo Er, and Chan Yee Chew. Aco for a new tsp in region coverage. In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1717–1722, 2005.

[28] Xiaoping Fan, Xiong Luo, Sheng Yi, Shengyue Yang, and Heng Zhang. Optimal path planning for mobile robots based on intensified ant colony optimization algorithm. In Proc. IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, volume 1, pages 131–136, 2003.

[29] Gengqian Liu, Tiejun Li, Yuqing Peng, and Xiangdan Hou. The ant algorithm for solving robot path planning problem. In Proc. International Conference on Information Technology and Applications , volume 2, pages 25–27, 2005.

[30] Xiaodong Zhuang, Qingchum Meng, and Bo Yin. Reinforcement learning with extended spatial and temporal learning scale. In Proc. IEEE International Conference on Tools with Artificial Intelligence , pages 324–328, 2003.

[31] Marco Dorigo and Luca Maria Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation , 1(1):53–66, April 1997.