The Seventeenth International Symposium on Artificial Life and Robotics 2012 (AROB 17th '12),
B-Con Plaza, Beppu, Oita, Japan, January 19-21, 2012

Motion planning using memetic evolution algorithm for network robot systems

Chien-Chou Lin[1], Wei-Ju Chuang[2], and Kun-Cheng Chen[3]

Department of Computer Science and Information Engineering,
National Yunlin University of Science and Technology,
Douliou, Yunlin 64002, Taiwan, R.O.C.

{[1]linchien, [2]g9917709, [3]g9817704}@yuntech.edu.tw

Abstract: A hierarchical memetic algorithm (MA) is proposed for the path planning of swarm robots. The proposed algorithm consists of a global path planner (GPP) and a local motion planner (LMP). The GPP plans a trajectory within the Voronoi diagram (VD) of the free space. An MA with a non-random initial population plans a series of configurations along the path given by the former stage. The MA locally adjusts the robot positions to search for better fitness along the gradient direction of the distance between swarm robots and intermediate goals (IGs). Once the optimal configuration is obtained, the best chromosomes are reserved as the initial population for the next generation. Since the proposed MA has a non-random initial population and local searching, it is more efficient and the planned path is faster than the traditional genetic algorithm (GA).

Keywords: memetic algorithm, genetic algorithm, hierarchical, local motion planner, swarm robots, Voronoi diagram

1 INTRODUCTION

In recent years, an increasing number of multi-robot systems have been proposed. Swarm robotics [1][2][3] is an approach for coordinating multi-robot systems. The swarm shares information about the environment and individual members interact with each other. Cooperative behavior may be used to complete a task. Most studies on robot swarm cooperation have focused on formation control, which refers to the task of controlling a group of mobile robots to follow a predefined path or trajectory while maintaining the desired formation pattern. Numerous methods have been proposed for formation control, which can be roughly categorized into four basic approaches, namely behavioral, virtual structure, leader-follower, and potential field.

In virtual structure approaches, the robot swarm is considered as a single rigid robot. A rigid geometric relationship among group members is maintained [4]. Therefore, the path planning of a robot swarm can be simplified as the path planning of a rigid robot. The advantage of the virtual structure approach is ease of implementation. However, the approach has low path planning flexibility.

For behavior-based approaches, several desired behaviors, i.e., movement towards the goal, obstacle avoidance, collision avoidance, and keeping formation, are defined for each robot to create its trajectory. The planning of robots can be done concurrently. Since each robot is considered individually, it is difficult to guarantee precise formation control.

In leader-follower approaches, the ability of a robot depends on its job. In the swarm, one or a few robots act as leaders which move along predetermined trajectories and other robots in the group follow while maintaining the desired relative position with respect to the leader. Generally, leader-follower-based robot systems are implemented as centralized systems. However, most leader-follower approaches are not complete algorithms because the safe path, that which gives a robot sufficient distance from obstacles and other robots is difficult to derive.

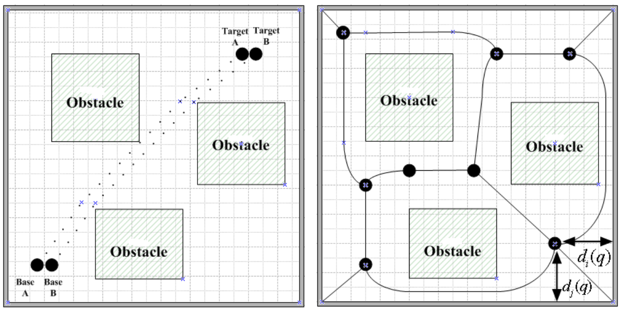
In order to obtain a safe path for swarm robots, the present paper proposes a hierarchical path planning algorithm. The proposed algorithm consists of a global path planner (GPP) and a local motion planner (LMP).

The rest of this paper is organized as follows. The GPP and the LMP of the proposed hierarchical path planning algorithm for swarm robots are introduced in Section 2 and 3, respectively. In Section 4, simulation results are given. Finally, conclusions and suggestions for future work are given in Section 5.

2 GLOBAL PATH PLANNING

Global path planning can be considered as a planning problem for a point robot. In Fig. 1(a), a swarm of two robots moves to the goal configuration; the planned path is close to obstacles [5]. In order to obtain a safe path, a Voronoi diagram (VD) is adopted since it is easy to implement and has been shown to work well in many cases. There are many variants of VD [6][7][8]. In the present study, a VD consisting of line segments is considered. A VD shows a set of free points which are equidistant to two closest obstacles. In [9], the VD was constructed using

Voronoi vertices and Voronoi arcs. The Voronoi vertices are points equidistant to the closest features of three (or more) polygons. The vertices are connected by continuous chains of Voronoi arcs. An arc may be equidistant to two closest vertices or to two closest obstacle edges or to an obstacle vertex and an obstacle edge.

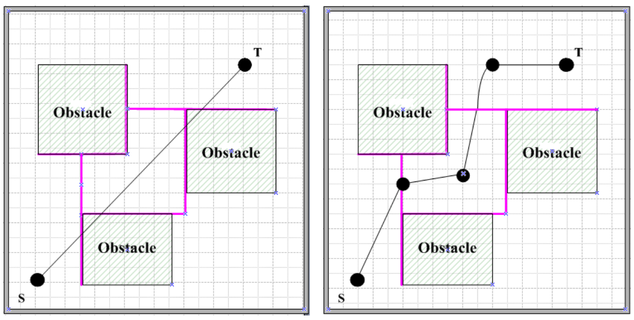


(a) (b)

Fig. 1. (a) Simple path of moving straight from start to goal. (b) Voronoi graph for Fig. 1(a).

As shown in Fig. 1(b), all edges and vertices of obstacles are used to construct the VD. The computation complexity is proportional to the total number of features of obstacles. Only a partial VD is used for global path planning for swarm robots. An efficient approach for constructing the partial VD is proposed in this paper.

Unlike approaches which construct the whole Voronoi diagram of the free space and then search for the path, the proposed scheme constructs a partial VD of the region of interest. As shown in Fig. 2(a), the proposed approach explores Voronoi vertices constructed from obstacles which are near the straight line from start to goal. Then, the Voronoi vertices are connected by a Voronoi arc which is formed by the nearest edges along the line, as shown in Fig. 2(b). The approach significantly reduces the computation complexity. Since a VD is the medial axis of the free space, the global path derived using a VD for swarm robots is the safest path.



(a) (b)

Fig. 2. The Voronoi vertices are connected by a Voronoi arc which is formed by the nearest edges along the line.

3 LOCAL MOTION PLANNING

The global path obtained in the previous stage can be sampled as a series of positions, denoted as (q_1, q_2, q_3, …,q_n), which the center of the swarm robots should follow. These positions can be considered as the intermediate goals (IGs). For each position q_i, the memetic algorithm (MA)-based local motion planner plans a set of configurations for the robots to which the center of swarm is fixed at point, q_i. The proposed memetic algorithm is:

Potential-based Memetic Algorithm
Begin
***i* = 1; /* Initialize the first intermediate goal */**
***t* = 0; /* Initialize the evolutionary generations */**
    Randomly generate an initial population *Pi (t)*;
    fitness(*Pi(t)*);
      repeat until (reach the final goal *qn*) Do
      Pi+1(t)= Pi(t);
        repeat until (reach the intermediate goal *qi*)
        Do
          select *Pi(t+l)* from *Pi(t)*;
          crossover(*Pi(t+l)*);
          mutate (*Pi(t+l)*);
          fitness(*Pi(t+l)*);
          apply FT Local Search to *Pi(t+l)*
          t = t +1;
        end
      i= i +1;
    end
  End

To apply an MA search for the optimal configurations, the coordinates of the robot swarm are encoded into one chromosome. The configuration of k robots is defined as their displacements, denoted as ((x_1, y_1), (x_2, y_2) ,…,(x_k, y_k)).

3.1 Initialization

The population, $P_1(0)$, of the first intermediate goal is generated randomly. The initial populations, ($P_i(0)$, i >1), of other intermediate goals are partially obtained from the last generation of the preceding intermediate goal and are partially randomly generated. Since these initial populations are eugenic and inherit from ancestors, the evolution time is reduced.

3.2 Natural selection

Natural selection is a genetic operator that chooses a chromosome from the current generation's population for inclusion in the next generation's population. Before making it into the next generation's population, selected chromosomes may undergo crossover and/or mutation (depending upon the probability of crossover and mutation) in which case the offspring chromosome(s) are actually the ones that make it into the next generation's population.

The aim of selection is to preserve optimal chromosomes and abandon suboptimal ones. In this study, the top percent scheme is adopted. The top 10 percent of the population is reserved as the next generation's population and the others are selected randomly.

3.3 Crossover and mutation operators

Selection alone cannot generate any new chromosomes for the population. The reproduction operators, crossover and mutation, are used to generate new offspring for the next generation. Crossover is performed between two selected chromosomes, called parents, by exchanging parts of their genomes to form two new chromosomes, called offspring. The most popular types of crossover operations are one-point, two-point, uniform, and blending. In this paper, since the *i-th* gene of a chromosome represents the position of robot *i*, the crossover operator exchanges similarly positioned genes of a pair of chromosomes.

For the mutation operator, an arbitrary bit in a genetic sequence is changed with a probability. The purpose of mutation in evolutionary algorithm (EA) is as a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next while attempting to avoid local minima.

3.4 Fitness function

Generally, selection is conducted according to the fitness of every chromosome, where the fitness evaluation of the GA is an objective function for chromosomes.

The fitness function can be rewritten [10] as:

$$V_q = f_{collide}(q) \times \left(\sum_{i=1}^{k} D^i{}_q + \rho U_{rep}(q) + \frac{1}{2} k \sum_{i=1}^{n} X_i^2 \right) \tag{1}$$

where $D^i{}_q$ is the distance between robot i and the intermediate goal of the swarm center, q. The ρ is a constant and $U_{rep}(q)$ is the repulsive potential of swarm robots from obstacles. When a configuration collides with obstacles, the collision function, $f_{collide}(q)$, is equal to V_{max}, which is a penalty; otherwise, it is equal to 1. The potential $U_{rep}(q)$ can be calculated analytically as:

$$U_{rep}(q) = \sum_{i=1}^{k} U^i_{rep}(q) \tag{2}$$

where $U^i{}_{rep}(q)$ is the repulsive potential of robot i from the nearest obstacle.

$$U^i{}_{rep}(p) = \begin{cases} \frac{1}{2}\eta \left(\frac{1}{Dist(p)} - \frac{1}{Q^*} \right)^2, & Dist(p) \le Q^* \\ 0, & Dist(p) > Q^* \end{cases} \tag{3}$$

where Q^* is the minimum distance from obstacles and η is a gain of the repulsive gradient. *Disp (i)* is the distance between robot *i* and the closest obstacle. For swarm cohesion, a robot in the swarm should keep a certain distance from the swarm center and not stray far from other robots. In the proposed algorithm, a spring function is adopted as a repulsive/attractive potential function in the fitness function. In (1), X_i^2 is the difference between the distance between robot *i* and the nearest neighbor robot and the safe distance which should be kept between robots.

3.5 Local search of memetic algorithm using gradient between swarm robots and IGs

Conventional genetic algorithms (CGAs) don't have a fine-tuning (FT) process to get closer to optimal solutions. Unlike CGAs, an MA is an EA with a local search process to refine individuals. In this paper, the local research scheme is used to adjust the position of the center of the robot swarm moving toward the IG. The top 20 percent of chromosomes are reserved for the next generation. The chromosomes are fine-tuned before the next evolution. Consider the position of the swarm center of a chromosome as $R^c = (R^c_x, R^c_y)$ and the current intermediate goal as $IG_i = (IG_{ix}, IG_{iy})$. The best movement direction is defined as:

$$\vec{p} = IG_i - R^c = \left(IG_{ix} - R^c_x, IG_{iy} - R^c_y \right) \tag{4}$$

Therefore, the fine-tuning procedure of the genes of the chromosomes is defined as:

$$gene_i = \begin{cases} gene_i + p_i, & if \quad gene_i + p_i \le gene_{\max} \\ gene_{\max}, & if \quad gene_i + p_i > gene_{\max} \end{cases} \tag{5}$$

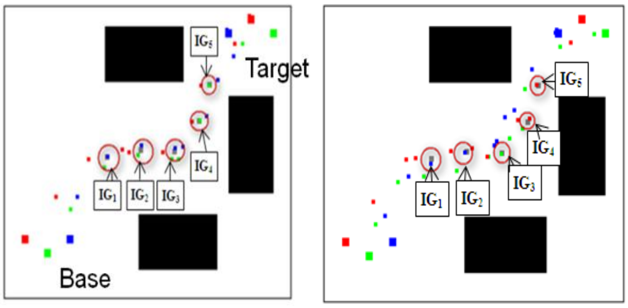
4 SIMULATION RESULTS

The proposed algorithm consists of the GPP and the LMP. The former was implemented using a modified Voronoi algorithm. The population was 100 and the maximum number of generations was set to 120. The probabilities of mutation and crossover were both 10%. The safe distances, *l* and Q^*, were set to 10 pixels and 2 pixels, respectively. The range of the genes was 30 to -30.

4.1 Swarm robot path planning by genetic algorithm and memetic algorithm

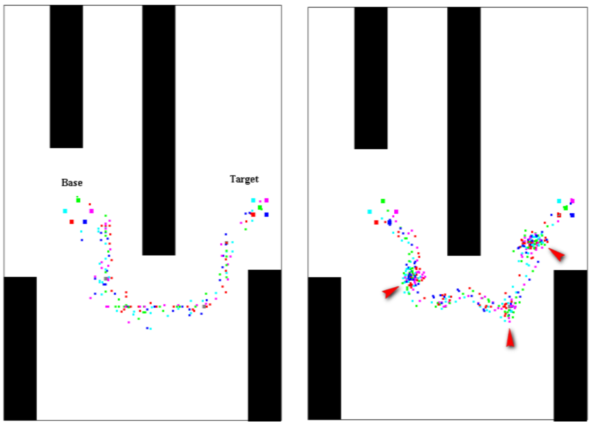
Case 1

The 3-robot swarm is shown in Fig. 3(a). There are 5 IGs and 3 obstacles in this case. The simulation took 8.609 seconds to plan a 10-configuration collision-free path. A similar simulation of the CGA is shown in Fig. 3(b). The CGA took 10.11 seconds to plan a 12-configuration path. The proposed algorithm is faster and more efficient.



(a) (b)

Fig. 3. Three trajectories for 3-robot swarm example. Motion planning obtained using (a) MA and (b) CGA.



(a) (b)

Fig. 4. Five trajectories for 5-robot swarm example. Motion planning obtained using (a) MA and (b) CGA.

Case 2

The simulation of a 5-robot swarm is shown in Fig. 4(a). The planned path is smooth. There are 4 IGs and 4 obstacles in this case. The simulation took 56.094 seconds to plan a 39-configuration collision-free path. A similar simulation of the CGA is shown in Fig. 4(b). The simulation took 88.563 seconds to plan a 59-configuration path.

5 CONCLUSION

The proposed MA has a non-random initial population and fine-tuning based local searching, which make it more efficient and faster than the traditional CGA. The proposed hierarchical approach avoids becoming trapped in local minima. The path planning problem for swarm robots was considered for 2-D workspaces. The proposed algorithm can be extended to 3-D workspaces without significant modification. For example, the gene of a robot can be represented as (x, y, z). In future works, we will focus on smoothing the planned paths to reduce redundant movements. With this modification, the proposed algorithm should be more efficient.

Acknowledgment. This work was supported by the National Science Council of Taiwan under grants NSC 98-2220-E-224-003, NSC 99-2220-E-224-007 and NSC 100-2218-E-224-009-MY3.

REFERENCES

[1] Yogeswaran M and Ponnambalam SG (2010), Advanced knowledge application in practice. Intech, Croatia, pp. 259-279

[2] Samitha WE and Pubudu NP (2009), Formations of robotic swarm: An artificial force based approach. International Journal of Advanced Robotic Systems 6(1): 7-24

[3] Alan FTW, Jin Sa, Mari CFG, Clare D and Michael F (2008), On formal specification of emergent behaviours in swarm robotic systems. International Journal of Advanced Robotic Systems, 2(4):363–370

[4] Beard RW, Lawton J and Hadaegh FY (2001), A coordination architecture for spacecraft formation control. IEEE Transactions on Control Systems Technology 9:777-790

[5] Ahmad G and SMRS N (2011), Optimal trajectory planning for design of a crawling gait in a robot using genetic algorithm. International Journal of Advanced Robotic Systems 8(1):29-36

[6] Choset H, Lynch KM, Hutchinson S, kantor G, Burgard W, Kavraki LE and Thrun S (1989), Principles of robot motion: Theory, Algorithms, and Implementations.

[7] Erinc G and Carpin S (2007), A genetic algorithm for nonholonomic motion planning. Proceedings of IEEE International Conference on Robotics and Automation, pp. 1843-1849

[8] Kloetzer M and Belta C (2007), Temporal logic planning and control of robotic swarms by hierarchical abstractions. IEEE Transactions on Robotics, 23(2):320-330

[9] Wein R, van den Berg JP and Halperin D (2005), The visibility-voronoi complex and its applications. In: Proceedings of the Twenty-first Annual Symposium on Computational Geometry, ACM, Pisa, Italy, pp. 63-72

[10] Lin CC, Hsiao PY and Chen KC (2010), A motion planning of swarm robots using genetic algorithm. Proceedings of the 2010 International Conference on Broadband, Wireless Computing, Communication and Applications, IEEE Computer Society, pp. 538-543