ROBOT PATH PLANNING IN UNSTRUCTURED ENVIRONMENTS USING A KNOWLEDGE-BASED GENETIC ALGORITHM

Simon X. Yang [*,**] **Yanrong Hu** [**]

[*] *Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, China*
[**] *School of Engineering, University of Guelph, Canada*

Abstract: In this paper, a knowledge-based genetic algorithm (GA) is proposed for dynamic path planning of a mobile robot in unstructured complex environments. A unique representation method is proposed for the 2-dimensional complex robot environment, where the obstacles could be of arbitrary shape. According to the problem representation, an effective evaluation method is developed specially for the proposed genetic algorithm. The evaluation method is capable of accurately detecting collisions between a robot path and an arbitrarily-shaped obstacle, and assigns a cost function that is effective for the proposed genetic algorithm. The proposed approach uses a problem-specific GA for robot path planning instead of the standard one. It incorporates the domain knowledge of robot path planning into its specialised genetic operators, some of which also involve a local search technique. The effectiveness and efficiency of the proposed genetic algorithm are demonstrated by simulation and comparison studies.

Keywords: Path planning, Mobile robot, Obstacle avoidance, Knowledge-based Genetic algorithm,

1. INTRODUCTION

Path planning is an important issue in mobile robotics. In an environment with obstacles, path planning is to find a suitable collision-free path for a mobile robot to move from a start location to a target location. Very often this path is highly desirable to be optimal or near-optimal with respect to time, distance or energy. Distance is a commonly adopted criterion. Robot path planning has been an active research area, and many methods have been developed to tackle this problem (Latombe, 1991), such as global C-space methods, potential field methods, and neural networks approaches. Each method has its own strength over others in certain aspects. Generally, the main difficulties for robot path-planning problem are computational complexity, local optimum, and adaptability. Researchers have always been seeking alternative and more efficient ways to solve the problem.

There is no doubt that path planning can be viewed as an optimisation problem (e.g., shortest distance) under certain constraints (e.g., the given environment and collision-free condition). Since the appearance of genetic algorithms (GAs) in 1975, GAs have been used in solving many optimisation problems successfully. It is not surprising that GAs are applied to path planning for mobile robots. However, like most early GA applications, most of those methods adopt classical GAs that use fixed-length binary strings and two basic genetic operators, and few modifications were made to the algorithms. Besides, most GA approaches for robot path planning deal with structured en-

vironments or environments with simple-shaped obstacles. In the genetic algorithms in (Wang *et al.*, 2002), obstacles do not have actual shape, and are usually represented by dots (for position only) or circles (for position and size). In some other genetic algorithms (Geisler and Manikas, 2002), environments with only rectangular obstacles are dealt with. In some approaches (Tu and Yang, 2003), grids and quad-tree are applied to the robot environments for discretisation, and obstacles are represented approximately by fitting into the cells.

The actual robot environments cannot be represented realistically when obstacle are represented as dots, circles, rectangles, or by fitting into cells. Real robot environments are usually unstructured and with arbitrarily shaped obstacles, so it is very important to represent the environment without distortion. Shibata and Fukuda (1993) proposed a genetic algorithm to deal with unstructured environments where obstacles are convex only. This approach is based on MAKLINK graph environment representation (Habib and Asama, 1991), and needs to form a configuration space before applying the genetic algorithm, which can be very time consuming. Besides, the paths are restricted to passing through the predetermined nodes of the graph. When dealing with unstructured environments with non-convex obstacles, evaluation problem becomes challenging, which requires accurate collision detection between paths and obstacles and effective quality assessment.

In this paper, a knowledge based genetic algorithm for path planning of a mobile robot in unstructured environments is proposed. The proposed GA uses problem-specific genetic algorithms for robot path planning instead of the standard GAs. The algorithm uses a unique problem representation method to represent 2-dimensional robot environments with complex obstacle layouts and obstacle are allowed to be of arbitrary shapes. In accord with the problem representation, an effective evaluation method is specifically developed for the proposed genetic algorithm. The evaluation method is able to accurately detect collision between a robot path and an arbitrarily shaped obstacle, and assigns costs that is very effective for the proposed genetic algorithm. The proposed knowledge based GA incorporates the domain knowledge into its specialised operators, some of which also combine a local search technique.

2. THE PROPOSED KNOWLEDGE BASED GA FOR PATH PLANNING

The proposed genetic algorithm features its simple and unique problem presentation, its effective evaluation method and its knowledge based genetic operators.

2.1 Problem Representation

Representation is a key issue in the work of GAs. The proposed genetic algorithm uses combination of grids and coordinates for problem representation. Grids function differently in this representation. The mobile robot environment is represented by a 2-dimensional continuous workspace, where obstacles are represented by the actual coordinates of their vertexes. A potential robot path is formed by several line segments connecting the start point S, intermediate nodes, and the target point T, where S and T are represented by their natural coordinates. An intermediate node is a node falling on one of the grids applied on the workspace. According to a certain resolution, the grids assigned with sequencing integer numbers are applied to the workspace to form the intermediate nodes of a path. The grids in this representation function differently from other grid-based methods. The grids here are *not* to discretize the whole environment and do *not* affect obstacle representation. Therefore, there is no distortion of the environment and obstacles. In fact the grids here indicate the resolution that only affects the intermediate nodes, and at the same time make it possible to use integers to represent a node instead of real-valued coordinates. Therefore, the chromosome structure and the genetic operations are simplified, and thus speed up the computation. Using grid numbers to represent intermediate nodes is acceptable as long as the resolution is high enough for the environment in question. Fig. 1 gives an example of an environment and path representation. Notice that the obstacles can be of arbitrary shapes and do not need to fit into the grids. Also, the start point and the target point are not restricted to any grids, and they can be any location in the environment. The resolution of the grids only relates fine-tunning of intermediate nodes. A feasible robot path is a collision-free path, i.e., none of the line segments intersect any obstacles. The length of a chromosome is variable, between the minimum of 2 and the maximum length $N_{\max}$. An example of path encoding is shown in Fig. 2, where S and T are not represented by grid numbers.

2.2 Evaluation

A robot path generated by the GA can be either feasible (collision free) or feasible. The evaluation should be able to distinguish feasible and infeasible paths and tell the difference of path qualities within either category. Particularly, it is very important for the proposed genetic algorithm to distinguish qualities of infeasible paths. The genetic algorithm generates its initial solutions randomly, and evolves solutions starting from the

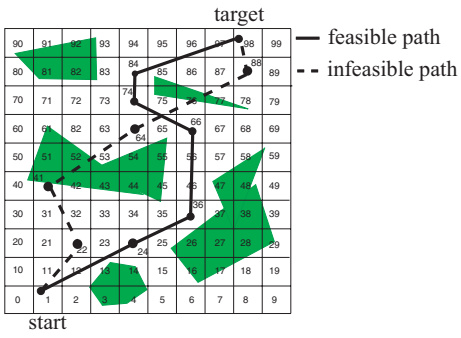

Fig. 1. A mobile robot environment and path representation example.

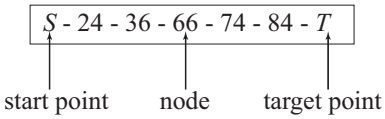

Fig. 2. An example of chromosomes in the improved GA

initial population. For an environment with many obstacles, it is very likely that the portion of feasible solutions in its population is small. Therefore, the GA needs to evolve feasible solutions from infeasible solutions. It is beneficial for the GA to evaluate the infeasible solutions that are easier to be evolved to feasible solutions as having better qualities.

The evaluation method should first check the feasibility of a path, which can be done by detecting collision between line segments of a path and obstacles in the robot environment. Then if the path is collision free, the length of the path is assigned as its cost indicating the quality. If the path collides obstacle(s), the evaluation method assigns the cost by estimating how deep the path intersects the obstacle, i. e., how difficult the path can escape the obstacle so that new solutions can be easily evolved from those easier-to-escape solutions. Based on analysis above, a evaluation method is given, and the evaluation function is defined as

$$F_{Cost} = \sum_{i=1}^{N} (d_i + \beta_i C), \tag{1}$$

where N is the number of line segments of a path, d_i is the Euclidean distance of the two nodes forming the line segment, C is a constant, β_i is the coefficient denoting depth of collision, and its definition is given as

$$\beta_i = \begin{cases} 0 & \text{if } i\text{th line segment is feasible} \\ \sum_{j=1}^{M} \alpha_j & \text{if it intersects obstacles} \end{cases} \tag{2}$$

where M is the number of obstacles the line segment intersects. α_j is determined by considering

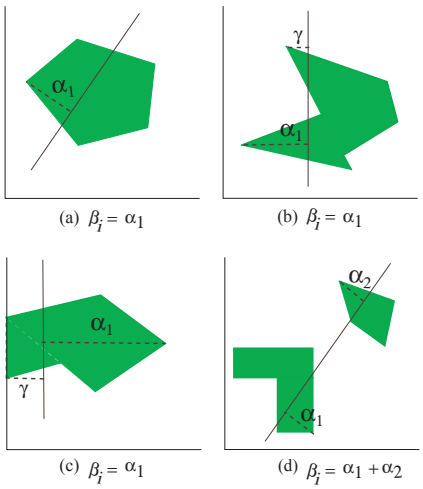

Fig. 3. Definition of the coefficient β_i.

how deep a line segment intersects an obstacle j. It is defined as the shortest moving distance for escaping the intersected obstacle. Fig. 3 explains the definition of α_j. In Fig. 3(a), α_1 is treated as the shortest distance to move the line out of the obstacle. In Fig. 3(b), γ is the shortest distance, but it is not enough to move the path away from the obstacle, therefore, instead α_1 is assigned to β_i. In Fig. 3(c), the obstacle is connected to the wall and is treated as a dead-end. The line can only escape from the other side of the obstacle. Thus β_i is assigned as α_1. Fig. 3(d) shows the situation that the line segment intersects two obstacles, it is considered to be more difficult for the path to move away from both, so the sum of α_1 and α_2 is used for β_i.

Accurate collision detection plays a key role in the evaluation. The method to check if a line segment intersects a convex obstacle was described by *Pavlidis* (Pavlidis, 1982). However, the proposed GA is dealing with arbitrarily shaped obstacles. Considering the specific requirements needed in the proposed GA on collision detection between a straight line segment and an obstacle as well as on quality assessment, we developed an algorithm for both collision detection and quality evaluation. An arbitrarily shaped obstacle is treated as a group of convex obstacles connected to each other. Thus, a robot environment consists of one or more groups of obstacles, each group consists of one or more convex obstacles that are called members of the group. A group information including how many members, which members, and its Min-Max Box of the group (MMBG) is obtained for each group. MMBG is calculated to indicate the minimal and maximal coordinates x and y of a group. Similarly, a Min-Max Box of Obstacle (MMBO) is also obtained for each member in a group. MMBG and MMBO play an important role to save computational time. A member information is also attached to each member, which

includes the number of vertexes, the coordinates of the vertexes, how many and which obstacles it connects and how it connects (connecting to its vertexes or sides), and its MMBO. With all the information, the algorithm is ready to evaluate a path. The path is checked with one line segment by another. First, the line segment is checked with each MMBG. If it does not intersect any MMBG, then it is collision free. If it intersects a MMBG, it needs to check with each member in the group to determine if there is actual collision. The line segment intersects a group if and only if it collides at least one member of the group. When collision is detected, the depth of intersection is calculated with the consideration of both group and member information. The process of assigning cost is actually the process of finding the vertexes from which the line segment escapes the obstacle most easily. Starting from the collided member of the obstacle, the process updates the easiest-to-escape vertex according to the connection status of members in the group and their status with regard to the line segment. Once the vertex is found, the shortest distance between the vertex and the line segment is used to calculate the cost.

The proposed evaluation method gives a penalty to infeasible paths, but still keeps them in the population pool because they might become good feasible solutions after certain genetic transformations. Importantly, this evaluation may allow some overlap between fitnesses of feasible and infeasible solutions by adjusting C in Eqn. (1). It would be beneficial to give more chance to some good infeasible solutions that would be easily to be evolved to good solutions. During the evaluation, some information obtained by the evaluation needs to be recorded so that later it can be used by some specialised genetic operators as heuristic knowledge without re-calculation in order to save computation time. The information includes feasibility (feasible or infeasible), number of infeasible line segments, and which obstacles intersected by which line segments of a path.

2.3 Genetic Operators

Those two commonly-used basic genetic operators, crossover and mutation, are not applicable for the robot path-planning problem here. They have to be tailored to suit for the problem and the adopted problem representation. In addition, to make the genetic algorithm more effective, three more specialised operators are designed to make use of available problem-specific knowledge, including the environment knowledge (e.g., numbers and positions of obstacles) and the path (e.g., feasibility and quality of a path). Some operators combine a small-scale local search technique. These five operators are illustrated in Fig. 4.

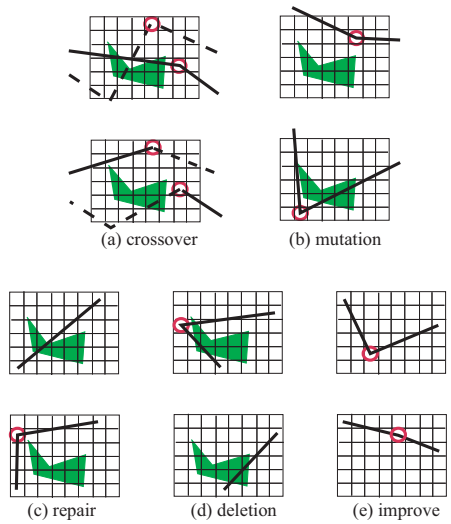

Fig. 4. Five specialised genetic operators that incorporates knowledge of robot path planing.

Crossover is the operator that randomly choose a node from Parent 1 and the other node from Parent 2. Exchange the part after these two nodes. Check these two offspring, and delete the part between two same nodes if it happens. The traditional 1-point or 2-point crossover cannot be used here because the length of a chromosome is variable. The choice of different crossover sites in different parents can increase the variability of chromosome length, which benefits exploration of the solution space.

Mutation is to randomly choose a node and replace it with a node that is not included in the path. Mutation is served as a key role to diversify the solution population. Therefore, it is not necessary that a solution is better after it is mutated.

Repair is to repair an infeasible line segment by inserting a suitable node between the two nodes of the segment. To locate the best node available, a local search is applied in the neighbouring girds of the intersected obstacle. When a node is not available to make the line segment feasible, a node that makes the line segment least infeasible is inserted. When the line segment cannot be improved by inserting a neighbouring node, a random neighbouring node is inserted, by which a line segment will not be stuck in any situation.

Deletion is applicable for both feasible and infeasible path. Randomly choose one node, check its two adjacent nodes and connected segments, if the deletion of the chosen node is beneficial (turn the infeasible to the feasible, or reduce the cost), delete the node.

Improvement is designed for feasible solutions. Randomly chose one node, do a local search in the neighbouring grids of the node, move to a better location. This operator is used for fine tuning of a feasible solution.

These operators are necessary to evolve feasible and good quality solutions. The firing of these operators depends on two criteria: probability and heuristic knowledge (e.g., if *feasible* then *improvement*). In an environment with many obstacles, the portion of feasible solutions in the initial population is small. *Crossover* and *mutation* operators are far from adequate to evolve good solutions. It is desirable to have these operators specially designed for robot path planning, such as *repair* and *deletion*, to evolve feasible solutions from the infeasible ones.

2.4 Outline of the Knowledge Based GA

An outline of the proposed algorithm is given in Fig. 5. Initial solutions are generated randomly and are evaluated by the fitness function in Eqn. (1). The population is evolved generation by generation. In each generation, selection and genetic operations are applied to the whole population. When two parents are selected according to some selection mechanism, one or more genetic operators are fired and applied to the parents to generate two children according to some probabilities and heuristic knowledge. After that, the whole population is replaced by the generated children with elitism. The use of elitism can prevent the GA from losing the good solutions found in the evolution and speed up convergence of the population. The best solution so far is updated in each generation, and it will be the final solution when some stop criterion is satisfied. The stop criterion can either be that the preset maximum generation is exceeded, or that the best solution remains unchanged for certain number of generations.

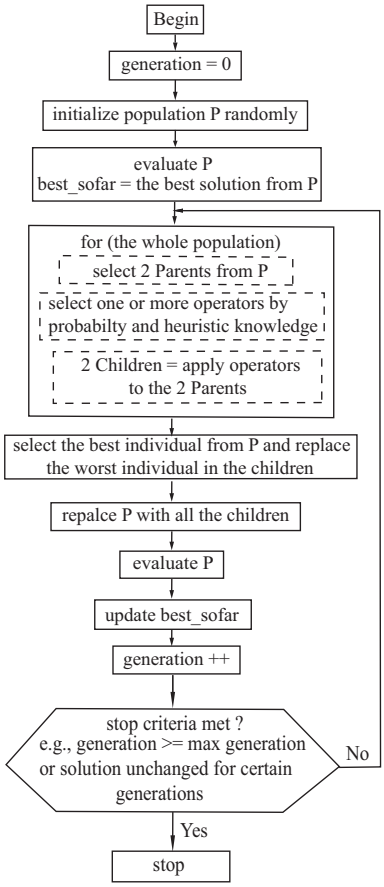

Fig. 5. Outline of the knowledge-based genetic algorithm for path planning of a mobile robot.

3. SIMULATIONS

To demonstrate the effectiveness of the proposed knowledge-based GA, several simulations were conducted. In the simulations, the parameters are set as: population size is 50, probability for mutation per chromosome is 0.2, and 0.9 for all the other operators. Tournament selection and elitism are applied. The proposed GA can deal with different resolutions. For simplicity, in all simulations, 100 unit $\times$ 100 unit grids is used unless it is otherwise stated. The simulation results also show that 100×100 resolution is high enough for the environments studied in this section. All simulations are conducted on a Pentium III PC (933 MHz) with Windows 2000.

3.1 Path Planning in Unstructured Environments

The proposed algorithm is first applied to a unstructured environment with many arbitrarily shaped obstacles. Fig. 6 shows one typical run. The algorithm first randomly generates the initial population (see Fig. 6(a)). Fig. 6(b) displays the best solution in the initial population, which is even infeasible and has the cost of 1043.25. Starting from this initial population, after selection and genetic operations, generation by generation, the population is evolved better and better. Fig. 6(c) shows the best solutions obtained from several different generations. It shows that the best solution first becomes feasible, and then the GA starts to improve the quality of solutions from each generation. The best solution in the population after certain number of generations is obtained as the final solution. Fig. 6(d) shows the finial solution obtained at generation 46. The cost is 117.56 and the computation time is 6.09 seconds. To test the robustness of the proposed GA, it has been run for many times. For 20 runs, the average cost is 117.91 with the standard deviation of 1.43, and the average computation time is 7.81 seconds with 2.58 standard deviation, and the average generations needed is 51 with 16 standard deviation.

3.2 Path Planning in a Clustered Environment

In this simulation, the proposed genetic algorithm is applied to a clustered environment with many

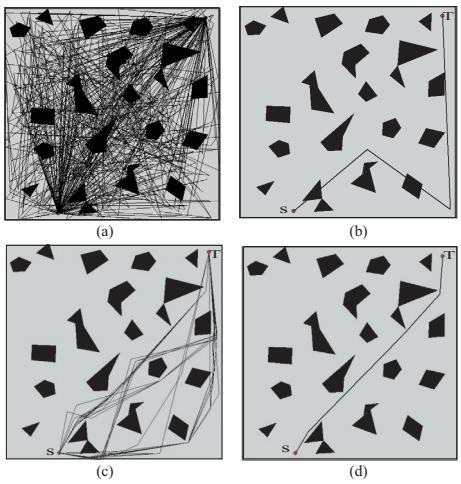

Fig. 6. One typical run of robot path planning in an unstructured environment. (a) the randomly generated initial paths; (b) the best path in the initial population; (c) the best paths from different generations; (d) the final solution path.

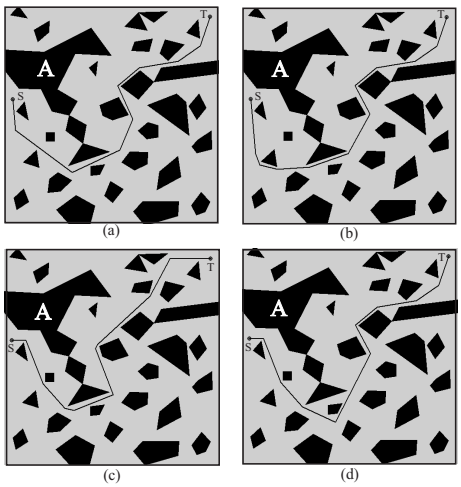

Fig. 7. Path planning in a clustered environment at four different runs.

arbitrarily shaped obstacle. Comparing to the above simulation, the coverage of obstacles in the environment greatly increases. Besides, it contains obstacles with very complicated shapes such as obstacle A in Figure 7. The simulation results in Figure 7 indicate that there is no problem for the GA to deal with those obstacles. some obstacles are connected only by a point, the GA can still accurately detect collision and no cut-through path is allowed. In a very complicated environment, there may be several near-optimal solutions. The GA is able to obtain different solutions in different runs because of the randomness involved in genetic algorithms. However, near-optimal solutions are guaranteed. Fig. 7 shows four near-optimal solutions from four different runs. Their costs are 160.50 (a), 160.93 (b), 163.95 (c) and 170.53 (d), respectively, which are slightly different, but they are all good solution paths to this complex environment. The average computation time for the four runs is 20.83 seconds.

4. CONCLUSION

In this paper, a knowledge-based genetic algorithm for path planning of mobile robots in unstructured environments is proposed. The GA uses a simple and unique path representation that uses natural coordinates to represent environment and grids to form the intermediate nodes of paths. The classical crossover and mutation genetic operators are tailored to the path planning problem. The proposed genetic algorithm also incorporates domain knowledge into its three problem-specific genetic operators for robot path planning. These operators also adopt small-scale local search based on some heuristic knowledge. The developed GA also features its one fitness function for both feasible and infeasible solutions. This evaluation method accurately detects collision between obstacles and paths, and effectively distinguishes qualities of both feasible and infeasible solutions, which is critical for the GA to evolve better solutions. The efficiency in computation time make the proposed knowledge-based genetic algorithm be able to be applied to real applications.

REFERENCES

Geisler, T. and T. W. Manikas (2002). Autonomous robot navigation system using a novel value encoded genetic algorithm. In: *Proc. of 2002 45th Midwest Symposium on Circuits and Systems*. pp. 45–48.

Habib, M. K. and H. Asama (1991). Efficient method to generate collision free paths for autonomous mobile robot based on new free space structuring approach. In: *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. pp. 563–567.

Latombe, J. C. (1991). *Robot motion planning*. Kluwer Academic Publisher. Boston.

Pavlidis, T. (1982). *Graphics and Image Processing*. Computer Science Press.

Shibata, T. and T. Fukuda (1993). Intelligent motion planning by genetic algorithm with fuzzy critic. In: *Proc. of Intl. Symposium on Intelligent Control*. pp. 565–570.

Tu, J. and S. Yang (2003). Genetic algorithm based path planning for a mobile robot. In: *Proc. of IEEE Intl. Conf. on Robotics and Automation*. Taiwan. pp. 1221–1226.

Wang, C., Y. Soh and H. Wang (2002). A hierarchical genetic algorithm for path planning in a static environment with obstacles. In: *Proc. of Canadian IEEE Conf. on Electrical and Computer Engineering*. pp. 1652–1657.