TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol. 12, No. 9, September 2014, pp. 6805 ~ 6810
DOI: 10.11591/telkomnika.v12i9.6271

Automated Ground Vehicles Navigation with Genetic Algorithm

Alireza Rezaee
Assistant professor of Department of system and Mechatronics Engineering,
Faculty of New sciences and technologies, University of Tehran,
P.O.Box 143951374, Tehran, IRAN
E-mail: arrezaee@ut.ac.ir

Abstract

Automated ground vehicles are being used increasingly in automation of factories for material transmitting or other missions. The navigation of these vehicles is a challenge as the configuration of the environment such as factory changes. The path-planning problem has been shown to be NP-hard, thus this problem is often solved using heuristic optimization methods such as genetic algorithms. In this paper a genetic algorithm for path planning of the mobile robots specifically automated ground vehicles with a basic knowledge about navigation area boundaries and configuration of obstacles. The goal in this paper is to travel the shortest path in minimal time while avoiding obstacles in a navigation environment. For this reason, an effective structure for genetic algorithm was implemented.

Keywords:

1. Introduction

Mobile robots are desirable for surveillance and operations such as bomb disposal or hazardous material management, which would be potentially dangerous for humans. Kind of industrial mobile robots being used in factory automation for material handling are automated ground vehicles (AGVs). An important task for these vehicles is autonomous navigation, where the vehicle travels between a starting point and a target point without the need for human intervention [1, 2]. While basic information may be available to the robot about the navigation area boundaries, unknown obstacles may exist within the navigation area. A path between the starting and target points that avoids collisions with obstacles is said to be feasible - this is a path that lies within free space. Thus, robot navigation methods need to solve the path-planning problem, which is to generate a feasible path and optimize this path with respect to certain criteria. Figure 1 illustrates a feasible path in a navigation environment from the top to the bottom.

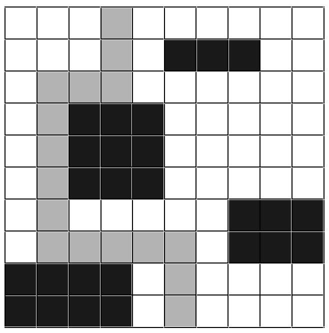


Figure 1. A Feasible Path in a Navigation Environment

Received May 15, 2014; Revised June 20, 2014; Accepted July 6, 2014

The goal for real-time mobile robots is to travel the shortest path in minimal time while avoiding obstacles in a navigation environment. Autonomous navigation allows robots to plan this path without the need for human intervention. As the path-planning problem has been shown to be NP-hard, this problem is often solved using genetic algorithm. An important part of the genetic algorithm solution is the structure of the genotype that represents paths in the navigation environment. The genotype must represent a valid path, but still be simple to process by the genetic algorithm in order to reduce computational requirements. Unfortunately, many contemporary genetic path-planning algorithms use complex structures that require a significant amount of processing, which can affect the real-time response of the robot. This paper describes the development of a genotype structure that contains only the essential information for path planning, which allows for more efficient processing. The genetic algorithm using this structure is evaluated on a variety of navigation spaces and is found to produce valid, obstacle-free paths for most cases.

Robot path planning is part of a larger class of problems pertaining to scheduling and routing, and is known to be NP-hard. Thus, a heuristic optimization approach is recommended [3]. One of these approaches is the use of genetic algorithms. A genetic algorithm (GA) is an evolutionary problem solving method, where the solution to a problem evolves after a number of iterations. A genetic algorithm starts with a population of chromosomes. Each chromosome represents a possible solution for a given problem. For robot navigation, a chromosome may represent a path between the starting and target points. Each chromosome is assigned a fitness value, based on how well the individual meets the problem objectives. Using these fitness values, individuals are selected to be parents. These parents form new individuals, or offspring, via crossover and mutation. Parent selection, crossover and mutation operations continue for several generations until the algorithm converges to an optimal or near-optimal solution.

Various genetic algorithm methods have been applied to the robot navigation problem. One approach is to combine fuzzy logic with genetic algorithms [4-6]. In this approach, the genotype structure represents fuzzy rules that guide the robot navigation, so the genetic algorithm evolves the best set of rules. While this approach can produce a feasible path through an uncertain environment, the genotype structure becomes very complex, as it needs to represent a variety of fuzzy rules. Another approach is to use genotype structures that represent local distance and direction, as opposed to representing an entire path [7-10]. While these are simple to process and allow for faster real-time performance, the local viewpoint of these methods may not allow the robot to reach its target. Some methods have relatively simple genotype structures that can represent feasible paths, but require complex decoders and fitness functions which can affect real-time response [11-13].

Our research is focused on improving the genetic algorithm performance by developing a simple and efficient genotype structure.

2. Genetic Algorithm Structure For Mobile Robot Navigation Problem Genotype Structure

The main point in solving the navigation problem is defining the genotypes structure to make it possible to solve the problem with genetic algorithm strategy. In order to simplify the navigation problem and defining the path which the robot negotiates, the environment of the robot is converted to rows and columns. The navigation model of the robot could be either row-wise or column-wise.

In row-wise navigation, as illustrated in Figure 2, the robot motion starts from one row toward another row passing rows consecutively. Given a navigation environment that is modeled by N rows, a path in that environment is represented by a genotype with N genes. Each gene position corresponds to a row index, while each gene value corresponds to a column index within that row. For example, the {1,7,4,4,5,8,6,2,3,10} chromosome in Figure 1, represents a path that starts in row 1, column 1 (1,1) and ends at row 10, column 10 (10,10). The intermediate points on this path are (2,7), (3,4), (4,4), (5,5), (6,8), (7,6), (8,2), (9,3), (10,10) respectively. Figure 2 shows the navigation along this path in the world space, where point (1,1) is assumed to be at the top left corner.

In column-wise navigation, in a chromosome, each gene position corresponds to a column index, while each gene value corresponds to a row index within that column. For example, the {7,5,4,8,10,6,5,3,7,8} chromosome in Figure 3, represents a path that starts in column 1, row 7 (7,1) and ends at column 10, row 8 (8,10). The intermediate points on this path are (5,2), (4,3), (8,4), (10,5), (6,6), (5,7), (3,8), (7,9), (8,10) respectively. Figure 3 shows the navigation along this path in the world space.

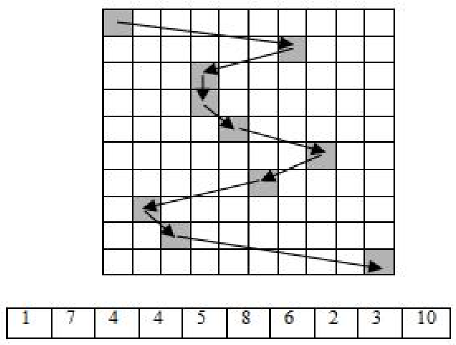


Figure 2. Row-wise Navigation and Related Coromosome

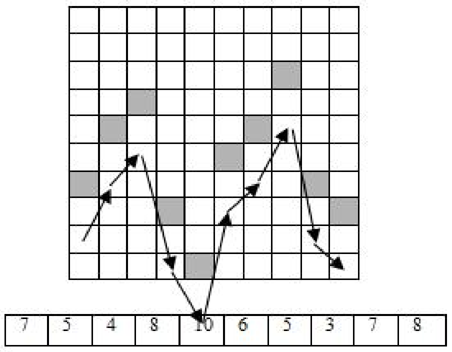


Figure 3. Column-wise Navigation and Related Coromosome

The direction information of a chromosome represents the intermediate steps of a path. However, sending the robot on a straight line directly from the center of one vertex to the center of the next vertex would mean that the robot moves on a diagonal line across many adjacent cells. This will cause problems if any adjacent cells that the robot traverses from one row to the next have an obstacle. A better approach is to split the diagonal path segment into a horizontal segment and a vertical segment, which will allow the robot to circumvent obstacles.

Assume that we have a segment that starts in row 1, column 2 (denoted by (1,2)) and ends in row 2, column 5 (2,5). If the direction bit is 0 (Figure 4), then the path segment is split into a vertical segment from (1,2) to (2,2) and a horizontal segment from (2,2) to (2,5). However, if the direction bit is 1 (Figure 5), then the path segment is split into a horizontal segment from (1,2) to (1,5) and a vertical segment from (1,5) to (2,5).

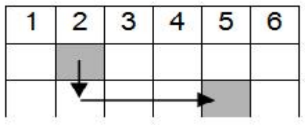


Figure 4. Vertical-horizontal Direction

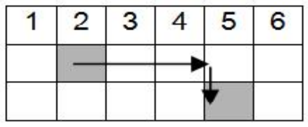


Figure 5. Horizontal-vertical Direction

For column-wise orientation, the direction bit is interpreted as follows: Assume that we have a column-wise path segment as shown in Figure 6. The segment starts in row 2, column 1 (2,1) and ends in row 5, column 2 (5,2). If the direction bit is 1 (Figure 6), then the path segment is split into a vertical segment from (2,1) to (5,1), and a horizontal segment from (5,1) to (5,2). Conversely, if the direction bit is 0 (Figure 7), then the path segment is split into a horizontal segment from (2,1) to (2,2) and a vertical segment from (2,2) to (5,2).

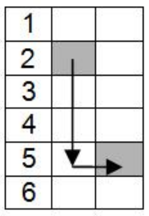


Figure 6. Vertical-horizontal Direction

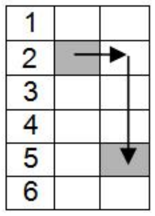


Figure 7. Horizontal-vertical Direction

Therefore, a direction bit is added to the chromosome structure to indicate the first direction that the robot will turn to proceed to the next vertex.

2.1. Crossover and Mutation

In our GA, two parent chromosomes are combined applying a single-cross-point, value encoding crossover. The crossover operator produces two offspring chromosomes with each crossover operation. During the operation of reproduction, crossover is applied on the chosen parent chromosomes only within the crossover probability. In the chosen crossover operator, two parent chromosomes are combined applying a single-cross-point, valueencoding crossover. The crossover operator has been modified to produce two offspring chromosomes with each crossover operation. This is achieved by using the gene information, which were not used to build offspring one, in order to build a second chromosome.

The chosen mutation operator checks with a mutation probability for every single gene whether it should be mutated or not. If a gene is to be mutated, a random number between 1 and the total number of columns in the search space is assigned to location and a random direction, either vertical or horizontal, is assigned to direction. This mutation variant has the advantage that it gives the opportunity for a chromosome to become significantly altered. That means that the complete search space will be explored and it therefore prevents the GA from getting stuck in a local optimum.

2.2. Fitness Evaluation

Each chromosome represents a path in the world space. Recall that the goal for autonomous robot navigation is to determine the shortest feasible path and traverse this path in minimal time. A path is evaluated by examining its segments. The total path length is the sum of its segments. If a segment intersects an obstacle, then this is called an infeasible step. Thus, a path is obstacle-free only if all of its steps are feasible. Minimizing path length will minimize travel time; however, so will minimize the number of turns required to traverse this path.

Given these criteria, our fitness function must consider feasibility, length, and number of turns as the fitness factors. The fitness function F is shown in Eq. (1) which includes S as the number of infeasible steps in the chromosome, L as the total length of the path segments and T as the total number of turns. K_f , K_L and K_T are weights for feasibility, length, and number of turns, respectively. After the fitness value for all chromosomes in the population have been computed, Rank Selection is used to determine the parent chromosomes that will be used for reproduction. By attention to this that the cromosomes with high fitness would be selected for next generartions, the fitness function is written in a way which increases with decrease of mentioned parameters.

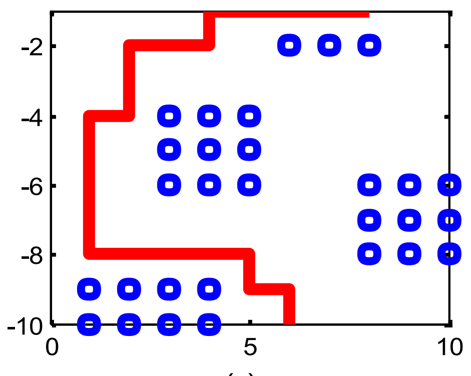
$$F = (K_f \times F_f + K_L \times L + K_T \times T)^{-1} \tag{1}$$

In this equation, by changing the weights, we could increase the value of the parameter which is more important for us in results. For example, avoiding obstacles is very important in robot navigation, so K_f is considered more than the other constants. In our simulations, the contants are considered as: $K_f = 100, K_L = 1, K_T = 1$.

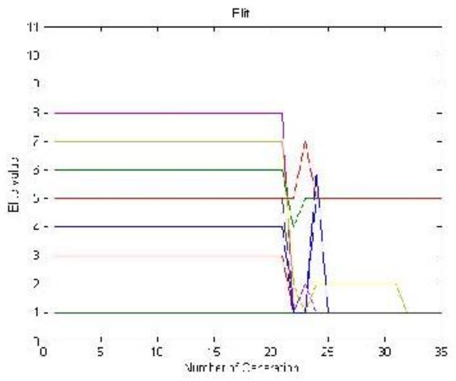
3. Simulation and Results

The genetic algorithm was applied to several sample spaces to simulate the navigation process. These spaces vary in size and in obstacle configuration. Several trials were run for each space. Crossover rate is 0.7, while mutation rate is 0.3. The population size is 30.

The numbers of generations depend on the complexity of the environment; consequently the generations increase as the environment space becomes large. Figure 8 and Figure 9 illustrate examples of successful runs for two sets.

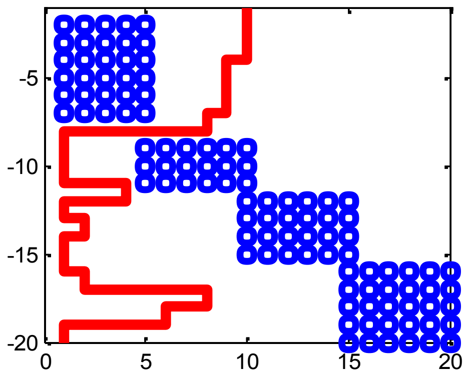


(a)

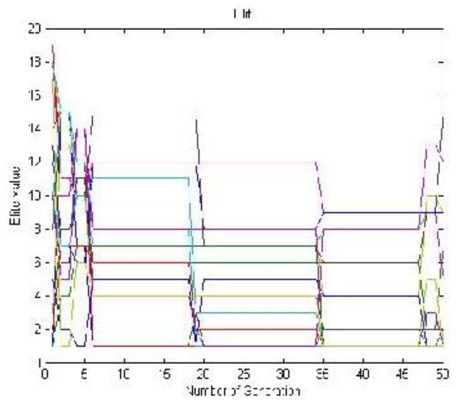


(b)

Figure 8. (a) result path for a 10×10 environment, (b) algorithm convergance



(a)



(b)

Figure 9. (a) result path for a 20×20 environment, (b) algorithm convergance

In our simulations, to get to better results, the fitness constants change as the environment size increase.

Our modification to the genotype structure to allow more options in path planning improves the success of robot navigation.

4. Conclusion and Suggestion

Simulations show the success of the genetic algorithm and the introduced approach in finding suitable path in navigation.

While the success rate was high, there were still trials where our GA failed to find a feasible path. Defenition of fitness function could effectively change the results. However defining suitable fitnesses independent of environment size toward problem goals could be very effective. New methods which combine row-wise and column-wise representations in the same genotype structure could be explored.

Another approch which will be our future work is implementing coevolution between pathes and directions in order to improve the success of the algorithm either in time or navigation performance.

References

[1] Shane Farritor, Steven Dubowsky. *A Genetic Algorithm Based Navigation and Planning Methodology for Planetary Robotic Exploration.* Proceedings of the 8th International Conference on Advanced Robotics. 1997.

[2] Prahlad Vadakkepat, Kay Chen Tan. *Evolutionary Artificial Potential Fields and Their Application in Real Time Robot Path Planning*. Proceedings on Congress on Evolutionary Computation. 2000.

[3] Hwang YK, Ahuja N. *Gross Motion Planning - A Survey*. ACM Computing Surveys. 1992; 24: 219-291.

[4] Arsene CTC, Zalzala AMS. *Control of Autonomous Robots Using Fuzzy Logic Controllers Tuned by Genetic Algorithms*. Proceedings of the 1999 Congress on Evolutionary Computation (CEC99). 1999; 428-435.

[5] Kubota N, Morioka T, Kojima F, Fukuda T. *Perception-Based Genetic Algorithm for a Mobile Robot with Fuzzy Controllers*. Proceedings of the 1999 Congress on Evolutionary Computation (CEC99). 1999; 397-404.

[6] Pratihar DK, Deb K, Ghosh A. *Fuzzy-Genetic Algorithms and Mobile Robot Navigation Among Static Obstacles*. Proceedings of the 1999 Congress on Evolutionary Computation (CEC99). 1999; 327-334.

[7] Cazangi RR, Figuieredo, M. *Simultaneous Emergence of Conflicting Basic Behaviors and Their Coordination in an Evolutionary Autonomous Navigation System*. Proc. 2002 IEEE Conf. on Evol. Comp. (CEC '02). 2002; 466-471.

[8] Di Gesu V, Lenzitti B, Lo Bosco G, Tegolo D. *A distributed architecture for autonomous navigation of robots*. Proceedings Fifth IEEE International Workshop on Computer Architectures for Machine Perception, 2000: 190 - 194.

[9] Gallardo D, Colomina O, Florez F, Rizo R. A Genetic Algorithm for Robust Motion Planning. 11th Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. 115-122.

[10] Vadakkepat P, Tan KC, Ming-Liang W. *Evolutionary Artificial Potential Fields and Their Application in Real Time Robot Path Planning*. Proc. 2000 Congress on Evolutionary Computation (CEC00). 2000: 256-263.

[11] Hocaoglu C, Sanderson AC. Planning Multiple Paths with Evolutionary Speciation. *IEEE Trans. Evolutionary Computation*. 2001; 5: 169-191.

[12] Sugihara K, Smith J. *Genetic Algorithms for Adaptive Motion Planning of an Autonomous Mobile Robot*. Proc. 1997 IEEE Int. Symp. Computational Intelligence in Robotics and Automation (CIRA '97). 138-143.

[13] Xiao J, Michalewicz Z, Zhang L, Trojanowski K. Adaptive Evolutionary Planner/Navigator for Mobile Robots. *IEEE Trans. Evolutionary Computation*. 1997; 1; 18-28.

[14] A Hermanu, K Ashenayi, TW Manikas, RL Wainwright. Autonomous Robot Navigation Using A Genetic Algorithm With An Efficient Genotype Structure. University of Tulsa, Tulsa, Oklahoma.

[15] Geisler T, Manikas TW. *Autonomous Robot Navigation System Using a Novel Value Encoded Genetic Algorithm*. 45th IEEE Int. Midwest Symp. On Circuits and Systems, Tulsa, OK. 2002; 45-48.