An Improved Robot Path Planning Algorithm

Xuesong Yan*1, Qinghua Wu2,3, Hammin Liu1,4
1School of Computer Science, China University of Geosciences, Wuhan, China
2Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, China
3School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China
4Wuhan Institute of Shipbuilding Technology, Wuhan, China
*corresponding author, e-mail: yanxs1999@126.com

Abstract

Robot path planning is a NP problem; traditional optimization methods are not very effective to solve it. Traditional genetic algorithm trapped into the local minimum easily. Therefore, based on a simple genetic algorithm and combine the base ideology of orthogonal design method then applied it to the population initialization, using the intergenerational elite mechanism, as well as the introduction of adaptive local search operator to prevent trapped into the local minimum and improve the convergence speed to form a new genetic algorithm. Through the series of numerical experiments, the new algorithm has been proved to be efficiency. We also use the proposed algorithm to solve the robot path planning problem and the experiment results indicated that the new algorithm is efficiency for solving the robot path planning problems and the best path usually can be found.

Keywords: Robot Path Planning, Genetic algorithm, Function Optimization, orthogonal initialization

1. Introduction

Autonomous robots are developed to perform high level task without further human operation. To accomplish these tasks the robots need to move in the real world. In consequence, one of the more important problems to be solved in the design of autonomous robots is the path planning. These planning problems can involve geometric workspace, constraints and additional complex features such as incomplete knowledge, moving obstacles, sensing and model uncertainties, unpredictability, kinematics, multiple robots and goals. In the last decade, path planning has received considerable attention from robotic communities since this fundamental operation requires the solution of a variety of theoretical and practical problems.

Until now, the research in robot path planning has focused on finding optimal routes from start to goal. The optimality is usually measured in terms of traveled distances [1]. Other measures are also used, e.g. confidence value [2]. For planetary rovers the efficiency of path is often expressed in terms of slope or roughness of the surface [3, 4].

Robot path planning is a NP problem [5], traditional optimization methods are not very effective to it, which are easy to plunge into local minimum. A lot of algorithms have been proposed to solve NP problem. Some of them (based on dynamic programming or branch and bound methods) provide the global optimum solution. Other algorithms are heuristic ones, which are much faster, but they do not guarantee the optimal solutions. The most popular evolutionary model used in the current research is Genetic Algorithms (GA), originally developed by John Holland [6]. The GA reproduction operators, such as recombination and mutation, are considered analogous to the biological process of mutation and crossover respectively in population genetics. The recombination operator is traditionally used as the primary search operator in GA while the mutation operator is considered to be a background operator, which is applied with a small probability.

Traditionally, GA uses a binary string representation of chromosomes with concentration on the notion of 'schemata'. A schema is a template that allows exploring the similarity among chromosomes. Genetic Algorithms model evolution as a search for structures or building blocks that perform well in a given environment. Therefore, the recombination and mutation operators focus on an individual's structure, not the structure's interpretation. The results of applying reproduction operation in GA generate solutions that share structural
similarities with their parents but may have significantly different interpretations. However, many recent applications of GA have used other representation such as graphs, Lisp expressions, ordered list, and red-valued vectors.

\[ t=0; \]
\[ \text{initialize } P(t) \]
\[ \text{evaluate structures in } P(t) \]
\[ \text{repeat} \]
\[ t= t+1 \]
\[ \text{select-reproduction } C(t) \text{from: } P(t-1) \]
\[ \text{combine and mutate structures in } C(t) \text{forming } C'(t); \]
\[ \text{evaluate structures in } C'(t) \]
\[ \text{select-replace } P(t) \text{ from } C'(t) \text{ and } P(t+1); \]
\[ \text{Until (termination condition satisfied).} \]

The above code gives the basic algorithmic steps for GA. After the initial population of individuals is generated (usually randomly) and individuals' structures are evaluated, the loop is entered. Then a selection buffer \( C(t) \) is created to accommodate the selected copies from \( P(t-1) \), "select-reproduction". In the Holland original GA, individuals are selected probabilistically by assigning each individual a probability proportional to its structural fitness. Thus, better individuals are given more opportunity to produce offspring. Next the variation operators (mutation and crossover) are applied to the individuals in \( C(t) \) buffer producing offspring \( C'(t) \). After evaluating the structural fitness of \( C(t) \), the selection method is applied to select replacement for \( P(t) \) from \( C'(t) \) and \( P(t-1) \).

In general, genetic algorithms are usually used to solve problems with little or no domain knowledge, NP-complete problems, and problems for which near optimum solution is sufficient. The GA methods can be applied only if there exist a reasonable time and space for evolution to take place.

2. Improved Genetic Algorithm

In our algorithm, the code representation we use the real coding, the real coding method has advantages compared with the binary coding in the function optimization problems, because the real coding can solve the "Hamming cliff" problem which the binary coding has no idea to solve it, and then the encoding, the decoding and the calculation of the fitness function is more convenient when we use the real coding.

2.1 Initialize Population

The traditional method of genetic algorithm is randomly initialized population, that is, generate a series of random numbers in the solution space of the question. Design the new algorithm, we using the orthogonal initialization [7] in the initialization phase. For the general condition, before seeking out the optimal solution the location of the global optimal solution is impossible to know, for some high-dimensional and multi-mode functions to optimize, the function itself has a lot of poles, and the global optimum location of the function is unknown. If the initial population of chromosomes can be evenly distributed in the feasible solution space, the algorithm can evenly search in the solution space for the global optimum. Orthogonal initialization is to use the orthogonal table has the dispersion and uniformity comparable; the individual will be initialized uniformly dispersed into the search space, so the orthogonal design method can be used to generate uniformly distributed initial population.

2.2 Intergenerational Elite Mechanism

Genetic algorithm is usually complete the selection operation based on the individual's fitness value, in the mechanism of intergenerational elite, the population of the front generation mixed with the new population which generate through crossover and mutation operations, in the mixed population select the optimum individuals according to a certain probability. The specific procedure is as follows:

- **Step 1**: Using crossover and mutation operations for population \( P1 \) which size is \( N \) then generating the next generation of sub-populations \( P2 \);
- **Step 2**: The current population \( P1 \) and the next generation of sub-populations \( P2 \) mixed together form a temporary population;
- **Step 3**: Temporary population according to fitness values in descending order, to retain the best \( N \) individuals to form new populations \( P1 \).
The characteristic of this mechanism is mainly in the following aspects. First is robust, because of using this selection strategy, even when the crossover and mutation operations to produce more inferior individuals, as the results of the majority of individual residues of the original population, does not cause lower the fitness value of the individual. The second is in genetic diversity maintaining, the operation of large populations, you can better maintain the genetic diversity of the population evolution process. Third is in the sorting method, it is good to overcome proportional to adapt to the calculation of scale.

2.3 Adaptive Local Search Operator

Local search operator has a strong local search ability, and then can solve the shortcomings of genetic algorithm has the weak ability for the local search. And the population according to the current state of adaptive evolution of the local search space adaptive local search operator will undoubtedly greatly enhance the ability of local search. In the initial stage of the evolution, the current optimal solution from the global optimum region is still relatively far away, this time the adaptive local search operator to require search a large neighborhood space to find more optimal solution, it can maintain the population diversity. When the population has evolved to the region containing the global optimum, the adaptive local search operator to require a relatively small area to search in order to improve the accuracy of the global optimal solution.

In our algorithm, the adaptive local search operator is the adaptive orthogonal local search operator. Adaptive orthogonal local search operator is aimed at the neighborhood of a point to search, so the key point is to identify a point as the center of the hypercube, the hypercube in the orthogonal test, expect to be better solution.

2.4 Framework of Algorithm

In this paper, we design the improved genetic algorithm with the real encoding, orthogonal initialization, combined with the roulette selection and the adaptive local search operator, using the intergenerational elitist mechanism [8,9]. The framework of the algorithm as follows:

Step 1 : Initialize a population P using the orthogonal initialization method, the scale is N, and evaluate the individuals in this population;

Step 2 : Using the roulette selection method, select two parent individuals from the population P respectively then using the crossover operation to get an optimal individual, then using the intergenerational elitist mechanism for this individual generate a sub-individual, and bring this individual into the new population pool P', repeat the step2 until the new population pool P' size is N;

Step 3 : Evaluate the new population P', then combined the new population P' with the populations P and form a temporary population;

Step 4 : Sort the temporary population according to the fitness value of the individuals in descending order; retain the best N individuals to form the next generation population P;

Step 5 : If not satisfy the termination condition, switch to step 2, otherwise, end of the algorithm.

2.5 Experiment Verify

In order to verify the improvement of the new algorithm, we use benchmarks [10,11] function to test the algorithm’s performance and compare the results with traditional genetic algorithm. Specific details of the test function see Table 1. In the Table 1, n behalf of the dimension number of the function, S behalf of the range of variables, \( f_{\text{min}} \) behalf of the minimization of the function.

The two algorithms of the same experimental parameters set. Each function in Table 1 is run 50 times with the two algorithms, their experimental results such as Table 2. By analyzing the experimental results we know, in solving function f1, f4 and f7, use the genetic algorithm is easily into local optimum, but use the new algorithm, convergence soon, and find a better solution, the average fitness and the best fitness was both superior to genetic algorithm. For the function f2, the new algorithm and GA all can find the global optimal, these two algorithm for this test function is very effective. For function f5, the two algorithms can find the best solutions are
the same (see Table 2), and the new algorithm to get the best value of the average is better than GA algorithm.

<table>
<thead>
<tr>
<th>Test Function</th>
<th>n</th>
<th>S</th>
<th>f_{\text{min}}</th>
</tr>
</thead>
<tbody>
<tr>
<td>f_{1}(x) = \sum_{i=1}^{n} x_{i}^{2}</td>
<td>30</td>
<td>(-100,100)</td>
<td>1495.71</td>
</tr>
<tr>
<td>f_{2}(x) = 6 \cdot \sum_{i=1}^{5} x_{i}</td>
<td>30</td>
<td>(-5.12,10.12)</td>
<td>0</td>
</tr>
<tr>
<td>f_{3}(x) = \sum_{i=1}^{n} i \cdot x_{i}^{2} + U(0,1)</td>
<td>30</td>
<td>(-1.28,1.28)</td>
<td>0</td>
</tr>
<tr>
<td>f_{4}(x) = \frac{1}{1000} \sum_{i=1}^{n} (x_{i} - 100)^{2} - \prod_{i} \cos \left( \frac{x_{i}}{\sqrt{i}} \right) + 1</td>
<td>30</td>
<td>(-300,0,300.0)</td>
<td>0</td>
</tr>
<tr>
<td>f_{5}(x) = -20 \exp \left( -0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_{i}^{2}} \right) - \exp \left( \frac{1}{n} \sum_{i=1}^{n} \cos \left( 2 \pi x_{i} \right) \right) + 20 + e</td>
<td>30</td>
<td>(-32.0,32.0)</td>
<td>0</td>
</tr>
<tr>
<td>f_{6}(x) = \sum_{i=1}^{n} 100(x_{i} - 10)^{2} + (x_{i} - 1)^{2}</td>
<td>30</td>
<td>(-2.048,2.048)</td>
<td>0</td>
</tr>
<tr>
<td>f_{7}(x) = \sum_{i=1}^{n} x_{i} \sin(\sqrt{1+x_{i}})</td>
<td>30</td>
<td>(-500,500)</td>
<td>-12569.5</td>
</tr>
</tbody>
</table>

In sum, we can see that in solving function f1, f4, f5 and f7, the new algorithm more efficient than GA algorithm, in solving other function, the performance almost same of the two algorithms. In short, this new algorithm has the following value: better global search capability.

3. Robot Path Planning Algorithm Based on Improved GA

We have chosen genetic algorithms to generate robot path mainly for two reasons: firstly, they are a powerful tool for searching in high dimensional spaces, like in this case. Secondly, the method imposes few mathematical constraints in the shape of the function to optimize. In this way, this method is applicable to the generation of a wide range of behaviors (obstacle avoiding, wall following, etc.).

3.1 Robot Path Planning Problem

Robot path planning can generally be considered as a search in a configuration space defined in what is known as the configuration space formulation [12-16]: Let A be a single rigid object, moving in a Euclidean space \( W = \mathbb{R}^{N} \), \( N = 2 \) or 3. Let \( O_{1}, \ldots, O_{n} \) be fixed rigid objects distributed in \( W \). The \( O_{i} \)'s are called obstacles. If \( A \) is described as a compact subset of \( W \), and the obstacles \( O_{1}, \ldots, O_{n} \) are closed subsets of \( W \), a configuration of \( A \) is a specification of the position of every point in \( A \) with respect to \( F_{W} \), where \( F_{W} \) is a Cartesian coordinate system. The configuration space of \( A \) is the space denoted by \( C \), with all possible configurations of \( A \). The subset of \( W \) occupied by \( A \) at configuration \( q \) is denoted by \( A(q) \). A path from an initial
configuration \( q_{\text{init}} \) to a goal configuration \( q_{\text{goal}} \) is a continuous map \( \tau : [0,1] \rightarrow C \) with \( \tau(0) = q_{\text{init}} \) and \( \tau(1) = q_{\text{goal}} \).

The workspace contains a finite number of obstacles denoted by \( O_i \) with \( i = 1, \ldots, n \). Each obstacle \( O_i \) maps in \( C \) to a region \( C(B_i) = \{ q \in C | A(q) \cap O_i \neq \emptyset \} \) which is called \( C \) obstacle. The union of all the \( C \) obstacles is the \( C \) obstacle region, and the set \( C_{\text{free}} = C \setminus \bigcup_{i=1}^{n} C(O_i) = \{ q \in C | A(q) \cap \bigcup_{i=1}^{n} O_i = \emptyset \} \) is called the free space. A free collision path between two configurations is any continuous path \( \tau : [0,1] \rightarrow C_{\text{free}} \).

The configuration space is a powerful conceptual tool because it seems to be the natural space where the path planning problem “lives.” This is mainly because any transformation of a rigid or articulated body becomes a point in the configuration space.

From a mathematical point of view [17], robot path planning problem can be expressed as:

\[
\min f(X), \quad X \in \mathbb{R}^n \\
\text{s.t.} g_i(X) \leq b_i, \quad i = 1, 2, \ldots, p
\]

where, the \( f(X) \) is the goal function, \( g_i(X) \) is the constraints condition and \( p \) denotes the numbers of the constraints in equation.

### 3.2 Path Representation

In our algorithm, to facilitate the coding of individual and genetic operation, the head point of the obstacles, the start point and the goal point are all called head point, give these head points the number, though a path can be denoted a head points sequence, which the start head point is \( v_1 \), and the goal head point is \( v_k \).

Each individual in the population represents a path starting from the initial point and trying to reach the final point. In this algorithm, we use list denote the path chromosome which can facilitate the insert and delete operation in the process of the path evolution. A single list denote a path, each list has a head point, all of the head point form the head array. So, the gene of the chromosome (the element of the list) denotes the point of the path. The Figure 1 is the chromosome representation of our algorithm.

![Figure 1. Chromosome representation](image)

Where, in the head point, \( \text{EleNum} \) denotes the number of the gene in the chromosome, \( \text{InterNum} \) denotes the number of the cross about the chromosome and the obstacles, \( \text{Length} \) denotes the length of the chromosome; where in each gene, \( \text{ObjNum} \) denotes the serial number of the obstacle, \( \text{PntNum} \) denotes the number of the gene in the obstacle \( \text{ObjNum} \), \( \text{Value} \) denotes the location of the gene in the head point sequence.

### 3.3 Fitness Function

Find fitness function is an important part of genetic algorithms. For the robot path planning problem, if the fitness function only consider the length of the individual path is not enough, taking into account the situation of the obstacles cross with individual paths. For this reason, in our algorithm fitness function measures the optimality of each path considering two factors: the distance between the final robot position and the goal point and the cross situation which the individual have no cross with the obstacles. Fitness function equation like following:
\[ f(S_i) = \frac{1}{\sum_{i=1}^{d-1} d(v_i, v_{i+1})} \]  

(2)

Where \( v_{d1} \) and \( v_{d2} \) is the location of the initial point \( s \) and the goal point \( g \) in the chromosome \( S_i \).

### 3.4 Algorithm Framework

Following provides a more detailed description of the whole robot path planning algorithm base on improved genetic algorithm.

Random initialization of the population \( P \)

initialize \( S(t) \)

\( S(t) = [S_1(t), S_2(t), \ldots, S_1(t)] \)

evaluate \( S(t) \)

\( f(S(t)) = \{ f(S_1(t)), f(S_2(t)), \ldots, f(S_i(t)) \} \)

While (not satisfied termination condition) do

for each individual \( S_i \in P \) do

begin

\( S^* \leftarrow S_i \)

Select (randomly) a Node \( C \) from \( S' \)

repeat

\{begin

if \((\text{rand}() \leq p)\)

select the node \( C' \) from the remaining nodes in \( S' \)

else

\{Select (randomly) an individual in \( P \)

assign to \( C' \) the next node to the node \( C \) in the select individual \}

\}

if (the next node or the previous node of node \( C \) in \( S' \) is \( C' \))

exit repeat loop

invert the gene node from the next node of \( C \) to the node \( C' \) in \( S' \)

\( C \leftarrow C' \)

\}

\}

if \((\text{eval}(S') \leq \text{eval}(S_i))\)

\( S_i \leftarrow S' \)

\end

end

\( i = i+1; \)

\end

PRINT \( X_{\text{best}}, f(X_{\text{best}}) \);

### 3.5 Experiment Result

We simulated the robot path planning algorithm with computer, and compared the planned path with the tradition method: visibility graph. For the two algorithms, we generated the obstacles randomly. Figure 2 is the visibility graph result, left corner is start point and right corner is intention point, the optimum path is shown with green line. Figure 3 is our algorithm result. From the result figures, the results indicated that our algorithm is efficiency for solving the robot path planning problems and it can found the right and optimal path. We also compared the calculation time with the two method, our algorithm is faster than the visibility graph.
4. Conclusion

This paper introduces a new algorithm based on the traditional genetic algorithm, for the traditional GA algorithm the new algorithm has done some improvements: By introducing genetic selection strategy, decreased the possibility of being trapped into a local optimum. Compared the traditional genetic algorithm, the new algorithm enlarges the searching space and the complexity is not high. By analyzing the testing results of nine benchmarks functions optimization, we reach the conclusion: in the optimization precision and the optimization speed, the new algorithm is efficiency than the traditional genetic algorithm and the new algorithm is more efficient than traditional genetic algorithm in coping with the function optimization problems. We also use this proposed algorithm for solving robot path planning problem and compared the experiment result with traditional method, the experiment results indicated that the new algorithm is efficiency for solving the robot path planning problems and the best path usually can be found.
Acknowledgments

This paper is supported by the Provincial Natural Science Foundation of Hubei (No. 2011CDB334 and 2011CDB346), supported by National Civil Aerospace Pre-research Project of China and the Fundamental Research Founds for National University, China University of Geosciences(Wuhan).

References