Optimal Robot Path Planning for Multiple Goals Visiting Using Tailored Genetic Algorithm

Fei Liu, Shan Liang*, and Xiaodong Xian

Abstract

During routine inspecting, mobile robot may be requested to leave for certain locations to perform particular tasks occasionally. This study aims at optimal path planning for this multiple goals visiting task based on tailored genetic algorithm. We employ two objectives, i.e., energy consumption and idle time (non-working time) [1-2]. Under constraint of energy, the proposed algorithm will generate an optimal path that comprises as more goals as possible and as less idle time as possible. In this algorithm, customized chromosome representing a path and genetic operators including *repair*, *cut* and *deletion* are developed and implemented. Afterwards, simulations are carried out to verify the effectiveness and applicability. Finally, analysis of simulation results is conducted and future work is addressed.

Key Words

Mobile Robot, Optimal Path Planning, Multiple Goals Visiting, Idle Time, Genetic Algorithm

1. Introduction

Mobile robots have been developed for many real-life tasks such as automatic patrolling in a transformer substation [3], welding automatically in a production line [4] and guiding in a campus [5]. For all the applications, path planning plays an important role in navigating robot to execute missions [6-7]. Further, it is generally occurred that more than one accessible path can be found, which indicates that a strategy is necessary for selecting the optimal or near-optimal path. In recent years, the important issue of optimal path planning has attracted considerable attentions.

Lab of Intelligent Sensing and Control, College of Automation, Chongqing University, China; emails: liufei2119@yahoo.cm.cn, {lightsun, xxd}@cqu.edu.cn

Manuscript received: 17, Nov. 2012 (write the date on which you submitted your paper for review.)

Types of Path Planning

Generally, path planning is to find a suitable collision-free path for a robot moving from a start point to a fixed target location [8-9]. In this situation, there have just one start location and one goal. However, in different applications, there are other four types of path planning according to the number of start points and goals: (i) one robot starts from a point, and chooses a goal from multiple candidate goals to move to. For example, when needing to recharge, the robot should select a docking station from all the stations to go to [1]; (ii) one robot moves from a start point and arrives at a destination while during this course, it must visit parts of the specified goals, for example, to pick up loads and carry them to the target location [10]. Specially, if the robot must visit all the path nodes in the environment and finally return to the initial point, it is well known as the routing problem of TSP (Traveling Sales Problem) [11]; (iii) multiple mobile robots leave from the same start point and go towards the same goal [12]; (iv) multiple robots start from different initial points and move to different goals [13-16]. In this study, we will solve the problem that a robot sets off from a point and traverses the specified goals. Compared with other researches, it has three properties: (i) none is designated as the ultimate goal; (ii) every goal may be the final destination according to (i); (iii) no order is set for visiting the goals.

Objectives for Optimal Path Determination

Among researches about optimal path planning, mainly path length is used for evaluating a path [17-18]. However, when various features of outdoor environment are considered such as friction and gravity, other criteria are proposed for determining an optimal path. For example, Wang et al. intend to plan a time-optimal trajectory for the mobile robot [19]. When finding optimal paths on terrains for a mobile robot, Sun et al. use energy consumed due to friction and gravity as the cost of a path [20]. Liu et al. also treat energy efficiency as the central factor in the cost function when developing the global path planner for the mobile robot [21]. In [22], researchers considered the energy expended on rotating, since in the environment with many walls and corners, it may cause much energy consumption if rotating frequently.

For planning an optimal path to multiple goals, Lobaton et al. took retracing into consideration [23]. In a particular application, the robot is required to pick up loads on the way, so the optimal way is that costing less time and collecting more loads [10].

In previous research on optimal path planning, on consideration of road attributes including length, road grade, surface roughness and the set of speed hump, we have studied optimal path planning based on energy consumption [24]. Further, by taking into consideration influence of vibration on mobile robot induced by motion, we proposed the decision factor---idle time (non-working time) as the cost of a path, which is proven to be more comprehensive on evaluating a path [1-2]. In this study both energy consumption and idle time are employed to evaluate the path.

Multiple Objectives Optimal Path Planning

Practically, when finding the optimal path, more than one factor will be taken into account. In terrains, as partial road segments are easy to pass while some are difficult, authors use two factors, namely path length and difficulty for evaluation of paths [25-26]. In parts of applications, there is no longer a single optimal solution but rather a whole set of possible solutions of equivalent quality if more than one objective is considered. When dealing with multiple objectives, researchers have created effective methods for different situations. A common way is to assign a weight to each objective, and then use the sum or product of the weighted value of each objective as the decision factor for evaluating a path [27-29]. The advantage of this method is simple and easy to realize. However in some situations, it is difficult to give an exact weight for each factor, so other methods dealing with the objectives are proposed. Takanori Shibata et al. use a fuzzy set to determine the fitness of a string (representing a path) where each value is decided by effects of two objectives, i.e., time and load [10]. Lexicographic method is another effective approach with which the objective functions are arranged in order of importance [30]. In addition, it is reasonable to pursue an optimal solution according to one criterion while satisfying other objectives that are used as constraints, which is called “Bounded objective function method” in [30]. Moreover, Gideon Avigad et al.

have studied the sequential optimization-constraint multi-objective problems, where different optimal solutions are generated in accordance with different planning demands [30].

GA-based Path Planning

Genetic algorithm (GA), based on the mechanism of natural selection and natural genetics, was first developed in the 1970s by Holland [32]. It is an evolutionary optimization method and is proven to perform well in optimal path planning [33]. To use GA, one should first find a pattern to express the feasible solutions, which is called chromosome. Besides, it is necessary to create a fitness function to evaluate each solution. The most challenging part is developing some appropriate genetic operators acting on the population of each generation that is the set of solutions. After evolving by certain generations, the optimal one will be determined by a criterion.

For diverse applications, due to the differentiation of problems, various modifications are made based on basic GA to solve concrete problems. In many occasions, researches use fixed-length chromosome to represent a path [34-35]. While in other circumstances, variable length chromosomes are used. For example, in a grid-based environment, authors use string of cells to describe a path whose length is unfixed [27, 36-38]. Meanwhile, different forms of fitness functions are created due to the fact that different objectives should be considered in respective application, such as path length [17], energy consumption [21], time consumption [19], smoothness and safety [39]. The key for evolution is the genetic operators. Traditionally, two operators, i.e., crossover and mutation are used nearly in all applications [37]. They play significant pole in adding diversity to the population and therefore are in favor of finding the global optimal solution. Apart from them, customized genetic operators are often established according to different purposes. For example, to make a feasible solution better, operator *improvement* is designed, which will randomly choose a node, and search in neighboring grids of the node, and move it to a better location [27]. This function is also realized in [25] by an operator with the name *repair*. In [27], there is also an operator called *repair*, while it is used for make infeasible path feasible. In

[38], *deletion* is employed to eliminate duplicate nodes existed in a path and adjust an individual in order of search direction. In addition, *insertion* is developed to make invalid path qualified by inserting nodes between unconnected nodes. Various customized operators have enlarged the field of application of GA-based method vastly.

In this paper, for the multiple goals visiting task, we proposed a tailored genetic algorithm to find an optimal path to visit as more goals as possible and two objectives, i.e., energy and idle time are both employed. As the robot is powered by rechargeable batteries, energy is limited and therefore should be taken into account in planning. In addition, on consideration of road attributes, we use idle time to evaluate a path if energy is sufficient. Based on the basic GA, modifications on expression of chromosome and genetic operators are made to fit the concrete case in this paper. This section has summarized related work and introduced our research. The remainder of this paper is organized as follows: in section 2, we will state the problem including the model of work environment, the multiple goals visiting task and properties of a path. In section 3, tailored genetic algorithm is described in detail. Then, simulations and analysis of results are conducted in section 4. Finally, conclusion and future work are addressed.

2. Problem Formation

In this section, we expand on the problem including the model of work environment and the task of multiple goals visiting. At last we introduce the properties of a path that are very important for the proposed genetic algorithm.

2.1 Model of Work Environment

We use a graph-based topological map to describe the work environment [1-2], which is illustrated in Fig. 1. In the map, $P_m (i.e.\ m = 0,1)$ represents a path segment, and A to H are nodes connecting two or more path segments respectively. For each segment, four attributes are considered, i.e., path length p_{ml} , surface

roughness p_{mr}, road grade p_{mg} and the set of speed-control hump p_{mh}. Besides, there have charging stations placed in the environment, each of which is named as $D_h (i.e.\ h = 0,1)$. For example, in Fig.1, D_0 is a charging station.

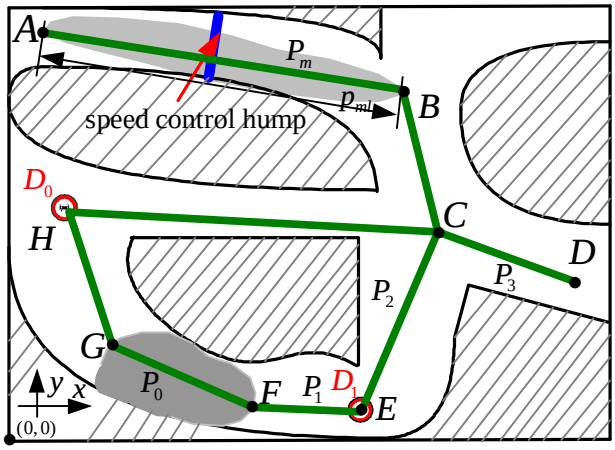

Figure 1. Model of work environment.

In previous study [1-2], we have elaborated the cost of a path segment. The cost that the robot will pay for passing each segment includes two parts: energy consumption c_e and the influence of vibration on robot body c_b. We use $C(c_e, c_b)$ to describe the cost of each segment. Furthermore, the calculation of c_e and c_b is

$$\begin{aligned} c_e &= \hat{c}_e + c_s \\ &= (r_e + r_s)\, p_{ml} \\ &= (\Delta r_{e\_r}\, p_{mr} \cos(p_{mg}) + \Delta r_{e\_g} \sin(p_{mg}) + r_s)\, p_{ml} \end{aligned} \tag{1}$$

$$c_b = \frac{r_{b\_\mu_f}}{\mu_f} \eta p_{mr}\, p_{ml} + p_{mh} r_{b\_h} \tag{2}$$

where $\hat{c}_e$ is the energy used for driving the motor and c_s is the energy that consumed by sensors on robot. By using c_e and c_b, the idle time is computed as

$$T_{IDLE} = f(c_e, c_b) = \frac{c_e}{v_{charge}} + c_b T_{MTTR} \tag{3}$$

where v_{charge} is the charging speed and T_{MTTR} is used to describe the Mean Time To Repair (MTTR) of our robot. Details of derivation of these formulations and descriptions of other parameters are not expected to shown in detail in this paper since they are available in previous work [1-2].

2.2 Task of Multiple Goals Visiting

Normally when executing regular inspecting task, the robot traverses in accordance with predefined route in the environment. Occasionally, the robot may be commanded to go to certain positions to perform particular mission. At this moment, it should stop and plan an optimal feasible path to visit the goals one by one to accomplish the task. For example, in Fig. 2, when the robot is at point S, it is asked to visit G_1, G_2 and G_3 temporarily.

The robot can choose any goal to visit first, which implies no order is set for the visiting task. Take the situation in Fig. 2 as an example. The robot can select the path coloured in blue to visit all the goals. Thus, the sequence of goals visited is $G_1 \rightarrow G_2 \rightarrow G_3$, for which we use $\Gamma_1 = \{S, A, G_1, G_2, D, G_3\}$ to describe the whole path for completing the task. However, the robot may choose visiting G_3 before G_2, then the sequence becomes $G_1 \rightarrow G_3 \rightarrow G_2$, and subsequently we get another feasible path $\Gamma_2 = \{S, G_1, G_3, G_2\}$ that is coloured in red in Fig. 2.

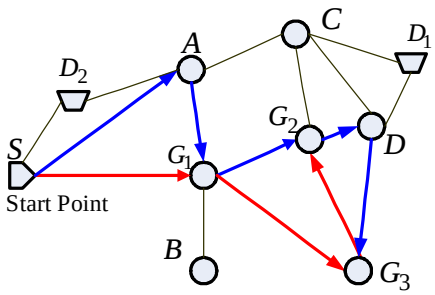

Figure 2. Task of multiple goals visiting.

The ideal situation is that the robot can visit all the goals by using the remaining energy. But in real-life cases, it is possible that the remaining energy is not sufficient for the robot to visit all the goals. Therefore, a compromising solution is to cut down a goal or more, and then try to find an optimal path for the visiting

task. The expecting result is to make the robot visit as more goals as possible under the precondition that after arriving at the last goal the robot still has enough energy to reach one charging station. For example, in Fig. 2, if the robot can't get to G_3 when following the most energy-saving path Γ_1, it can decide not visiting goal G_3, then the optimal path is $\Gamma_1 = \{S, A, G_1, G_2\}$ and G_2 becomes the last goal to be visited. Certainly the worst situation is that the remaining energy can not support for visiting even one goal. Consequently, three cases can be obtained that are shown in Fig. 3.

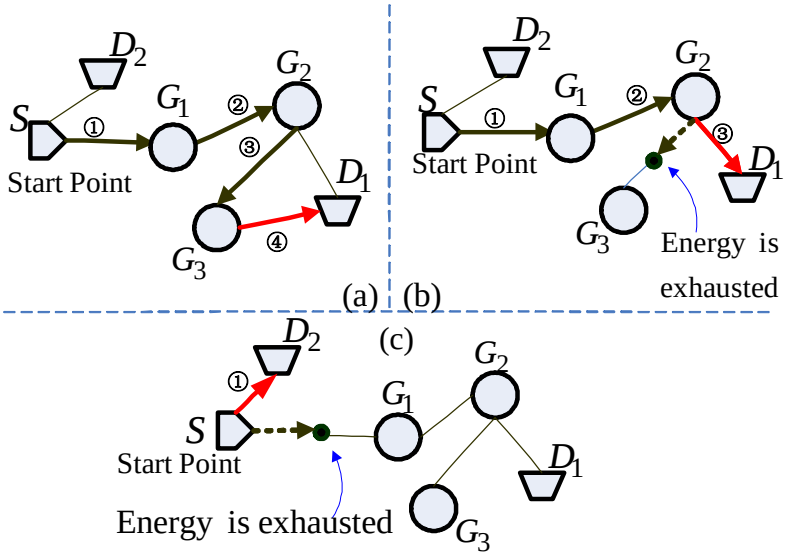

Figure 3. Three cases of task execution.

(i) The robot can visit all the goals and the remaining energy is enough for reaching a charging station, which is shown as situation (a) in Fig. 3;

(ii) The robot can just visit a portion of the goals and has to go back to one charging station. For example, in situation (b) in Fig. 3, the robot can not get to G_3 after visiting G_2 which means the robot can just visit two goals, i.e., G_1 and G_2.

(iii) The robot can not visit even one goal and has to go back to recharge. In situation (c) in Fig. 3, the robot even has not enough remaining energy to visit the first goal G_1. It should go back to charging station to recharge right now. Thus, it will not execute the task.

2.3 Properties of a Path

Several properties of a path are stated in this subsection. In the topological map used to describe the environment, the road network is constituted by connected nodes, which is illustrated in Fig. 1. Therefore, we use the combination of nodes to represent a path. In this research, four properties of a path are obtained:

(i) A path is constituted of parts of the nodes. For example, the path coloured in blue in Fig. 2 can be described as $\Gamma_1 = \{S, A, G_1, G_2, D, G_3\}$. This path is constituted by nodes S, A, G_1, G_2, D and G_3 in which G_1, G_2 and G_3 are the goals assigned.

(ii) There is no priority or constraint for the sequence of goals to be visited. The ultimate purpose is to visit as more as possible goals. So, whichever is visited first is permitted. For example, in Fig. 2, both paths $\Gamma_1 = \{S, A, G_1, G_2, D, G_3\}$ and $\Gamma_2 = \{S, G_1, G_3, G_2\}$ are admissible.

(iii) It is permissible for a node appearing in the sequence more than once. For instance, in Fig. 4, one available path is $\Gamma = \{S, G_1, S, G_2, G_4, G_2, G_3\}$, where S and G_2 appeared twice. The purpose of the first arrival at one goal is to perform task, and that of the other times are for going to other goals.

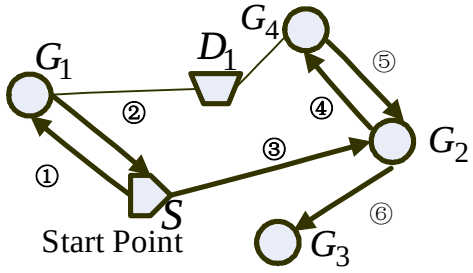

Figure 4. A special situation.

(iv) Since it may occur that the energy is not sufficient for visiting all the goals, the optimal path may not include all the goals. That is to say, for the case (b) shown in Fig.3, the optimal path will be $\Gamma = \{S, G_1, G_2\}$, which does not include goal G_3. And in situation (c) in Fig. 3, the robot can not visit any goal and therefore there is no feasible path. Consequently, it is not requested that the result of path planning has to be a feasible path including all the goals. It is also reasonable to report the result that no feasible path is possible for the robot performing the task.

3. Proposed Genetic Algorithm for Path Planning

In this section, we will first introduce the basic knowledge of genetic algorithm. Then the proposed tailored genetic algorithm is introduced to solve the problem described above.

3.1 Basis of Genetic Algorithm

As an advanced stochastic search technique similar to natural evolution based on the principle of "survival of the fittest" [27], traditionally, genetic algorithm involves three basic genetic operations to stimulate the adaptive process of natural systems: *Selection, Crossover and Mutation*.

Selection is an operator to select the survival in a set of present candidate individuals (usually being called *population*) according to the fitness value computed by the fitness function. The selected individual(s) will be kept surviving in the next generation.

Crossover is an operator adopted to reform the survival candidates. Usually, it is performed by exchanging parts of strings by use of old strings and then new strings are generated. This process derives from the natural system, in which a set of creatures creates a new set of the next generation by swapping among the creatures. Often the parts are crossed in couples of candidates selected randomly. When using this operator, one should determine a crossing rate to decide how often the selected individuals will carry out this operation.

Mutation is another way to increase the diversity of population. Compared to *crossover*, mutation rate is much smaller. Another important function of this operator is to avoid trapping in the local minima in the search space.

3.2 Tailored Genetic Algorithm

Based on traditional genetic algorithm, modifications are made to fit our problem. We use the combination of nodes to represent the chromosome. The fitness functions include two parts which are used to calculate

energy consumption and idle time respectively. Except the basic three operators, i.e., *selection*, *crossover* and *mutation*, we create three operators: *repair*, *cut* and *deletion*.

3.2.1. Chromosome

The proposed genetic algorithm uses the combination of nodes for path representation. An example of path encoding is shown in Fig. 5, which is $S-A-G_1-G_2-D-G_3$. In this chromosome, S is the start point, A and D are non-goal points, and G_1, G_2 and G_3 are three goals.

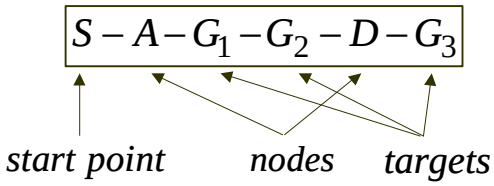

Figure 5. An example of chromosome in tailored genetic algorithm.

Two different chromosomes may have different length. For example, the length of the chromosome shown in Fig. 5 is 6, in which 3 goals are involved. While the length of the chromosome in Fig. 2, $S-G_1-G_3-G_2$, is 4, and the same 3 goals are included.

3.2.2. Evaluation

Chromosomes are selected for reproduction through genetic operators based on the fitness function, so it is important to establish a set of criteria to evaluate the quality of a path. For each individual, we adopt three terms Ω, F_e and $F_{T_{IDLE}}$ to describe it. Ω is the number of goals involved in this chromosome. Thus, $\Omega(\Gamma)$ represents the number of goals in path Γ. F_e is the energy that the robot will spend on moving along this path, and $F_{T_{IDLE}}$ indicates the idle time induced by this path. The total energy consumption of a path is the sum of that of each path segment, so

$$F_e = \sum_{i=1}^{N} c_e(i) \tag{4}$$

where N is the number of segments, $c_e(i)$ is the energy consumption of the ith segment.

Similarly, $F_{T_{IDLE}}$ can be calculated as

$$F_{T_{IDLE}} = \sum_{i=1}^{N} T_{IDLE}(i) \tag{5}$$

where N is the number of segments and $T_{IDLE}(\mathrm{i})$ is the idle time of the ith segment. $c_e(i)$ and $T_{IDLE}(\mathrm{i})$ are introduced in section 2.1.

Define E_{rest} as the available energy the robot can utilize to perform the task, then we get

$$E_{rest} = E_{cur} - E_{low} \tag{6}$$

where E_{cur} is the remaining energy the robot has when it is at the start point, and E_{low} is the threshold value of low energy.

We define e_{task} as the extra energy consumption that the robot spends on executing task at each goal, and assume that it is a fixed value, namely, $e_{task} = \tau$. For a path Γ, the total extra energy E_{task} spending on performing task is

$$E_{task} = \Omega(\Gamma) * e_{task} = \Omega(\Gamma) * \tau \tag{7}$$

Thus, we get another condition that makes one path feasible, namely,

$$F_e(\Gamma) + E_{task} \leq E_{rest} \tag{8}$$

which means that the sum of the energy spent on walking along the path and executing tasks should be less than the remaining energy.

3.2.3. Genetic Operators

In proposed genetic algorithm, except the three basic operators, i.e., ***selection***, ***crossover*** and ***mutation***, we create three other operators, i.e., ***repair***, ***cut*** and ***deletion***.

(1) Selection

The *selection* operation will select the best individual from the population in each generation and keep it in the next generation. The selection is based on the fitness value. Here, some special characteristics of this operator are emphasized in the following.

At the beginning of the genetic algorithm, candidate solutions are generated randomly, which constitute the initial population. We assume that there exists at least one feasible path Γ that involves all the goals and its energy consumption $F_e(\Gamma)$ meets equation (8), then in the initial population each candidate solution will include all the goals. For each generation, we will calculate the three terms for evaluating each path.

Initially, we first check if any individual meets equation (8). If at least one solution conforming to this condition is found, we will use $F_{T_{IDLE}}$ as the criterion to select the best one from individuals that meet this condition. Otherwise, if none is found, the selection process will utilize F_e to evaluate a path since the one having less energy consumption is more likely to be evolved to a feasible one. Subsequently, the best one that has the minimal F_e or $F_{T_{IDLE}}$ will be selected to remain in the next generation. This strategy can guarantee that the best one up to now will not be destroyed by other genetic operations and can accelerate the convergence of the algorithm.

(2) Crossover

Crossover is an efficient way to add diversity to the population. Firstly, a crossover probability is predefined. In this operation, two parents are selected randomly and a position is selected randomly too. Then, a random probability is generated. If the probability value is less than the predefined value, the operation will go on. Otherwise, the two parents are passed to the next generation directly. The operation will end until certain times of crossing operations are carried out.

When executing *crossover* operation, a crossover point will be generated. Since the length of two parents may not be the same, the sequence number of the point will not be bigger than the length of the shorter one. Then, in the other parent, we find the corresponding node and its sequence number of the first appearance.

If the other one has the same node, exchange the latter parts of the two parents. If not, quit and restart from choosing parents.

The following is an example of *crossover* operation. First, two parents are selected:

Parent 1: $S-G_1-G_3-G_2$

Parent 2: $S-A-G_1-G_2-D-G_3$

If point G_1 is selected as the position for exchanging, then we get the offspring after crossing:

Child 1: $S-G_1-G_2-D-G_3$

Child 2: $S-A-G_1-G_3-G_2$

After crossing, the two children are put into the population of next generation.

(3) Mutation

In *mutation* operation, a position is randomly chosen and the node at this position is replaced with a different node. *Mutation* is served as a key role to diversify the solution population. Therefore, it is not necessary that a solution is better after mutating. After mutating, this node may not be connected directly with the two nodes before and after. For example, if node A in path $S-A-G_1-G_2-D-G_3$ shown in Fig. 2 is chosen to mutate, and changes to C, then, this individual becomes $S-C-G_1-G_2-D-G_3$. However, as seen in Fig. 2, nodes S and C, and C and G_1 are not connected directly, which is to say, the individual after mutation is not a feasible solution. Even so, it has made the population diversified, and the following operator *repair* can make it feasible.

(4) Repair

When executing genetic operators, some infeasible paths may appear. For instance, after mutation, individual $S-A-G_1-G_2-D-G_3$ becomes $S-C-G_1-G_2-D-G_3$. When this happens, we will use *repair* operator to solve this problem. The practical way is inserting some suitable nodes between the two nodes.

Take string $S-C-G_1-G_2-D-G_3$ as an example. When executing ***repair*** operation, we first check if this individual is feasible by examine every two adjacent nodes. If at a position, the node and the next node are not connected directly, then, this operator will try to add some nodes between them in order to make the two connected reasonably. In the above example, the nodes S and C may be inserted by node A, and then C and G_1 may be inserted by nodes G_2 or A, which is decided randomly. If A is selected, then the individual is repaired to be $S-A-C-A-G_1-G_2-D-G_3$, and if G_2 is selected, it will become $S-A-C-G_2-G_1-G_2-D-G_3$. No matter whichever is chosen, the result is that the path becomes feasible at last.

It is notable that after mutation or crossover, the number of goals may change. For example, an individual should have k goals but just involve $k'(k'<k)$ goals. Then the repair operator will try to repair this string by adding nodes after the last node until the other k-k' goals are included at least one time.

(5) Cut

In a chromosome, it is admissible that any node appears more than one time. But the unnecessary reduplication must be avoided. For example, in the string $S-A-C-A-G_1-G_2-D-G_3$ obtained after repairing, node A appears twice and between them there has no goal. It can be regarded as that between the two times arriving at A, the intention is not for going to any goal. So, the sequence $C-A$ is meaningless and it needs to be cut. Finally, this string becomes $S-A-G_1-G_2-D-G_3$. Therefore, the ***cut*** operator is to do such things that cutting the unmeaning sequences existing in each individual.

However, the reduplication does not include the situation that a goal exists between the same two nodes. For instance, in chromosome $S-A-C-G_2-G_1-G_2-D-G_3$, goal G_2 appears twice. But between them there is another goal G_1 which indicates that the purpose of arriving at G_2 for the second time is for visiting another goal. Thus, the second time passing G_2 is meaningful.

From the above analysis, two criteria can be used as judging whether a chromosome needs ***cut*** operation:

(i) If a node that is not a goal appears twice, and no goal is in the sequence, then this sequence can be cut to the first node. For example, the sequence $A-C-A$ appearing in path $S-A-C-A-G_1-G_2-D-G_3$ can be cut to A.

(ii) In an individual, for each goal, only the first appearance is regarded as a goal. The other times of appearance will be regard as a common path node, which implies that the purpose of passing the goal for the second or third time is for reaching another goal. Take $S-A-G_1-A-\cdots-B-G_1-B-G_3$ as an example in which the robot reaches G_1 twice. Only the first time is for visiting this goal. This is why we cannot execute *cut* operation on the part A-G_1 - A. On the contrary, the second time when it appears, it will be deemed as a common node. Therefore, $B-G_1-B$ should be cut to be B according to rule (i).

(6) Deletion

In a chromosome, when all the goals have appeared once, the remainder of the string is not necessary and should be deleted. Assume that the string is admitted to include k ($0<k\le N$, N is the number of goals assigned at the beginning) goals, then if k' ($k'>k$) goals appear in the string, we should delete the string after the k th goal.

When it is confirmed that the robot has no sufficient energy to visit all the goals, it will attempt to reduce one goal and go on searching the optimal path. For example, in the task of visiting goals G_1, G_2 and G_3, if it is found that the robot has no sufficient energy to visit all the three goals, then the robot will try visiting two goals. Thus, the path $S-A-C-G_2-G_1-G_2-D-G_3$ will become $S-A-C-G_2-G_1$ after performing *deletion* operation.

3.2.4 Execution of Proposed Genetic Algorithm

Compared to traditional genetic algorithm, variations are made in the execution of proposed algorithm. It is cyclically executed, where the criterion for deciding the circulation is to check if any path satisfying the

energy constraint. In each cycle, the evolution process is similar to the basic GA. In Fig. 5, the execution of proposed GA is illustrated.

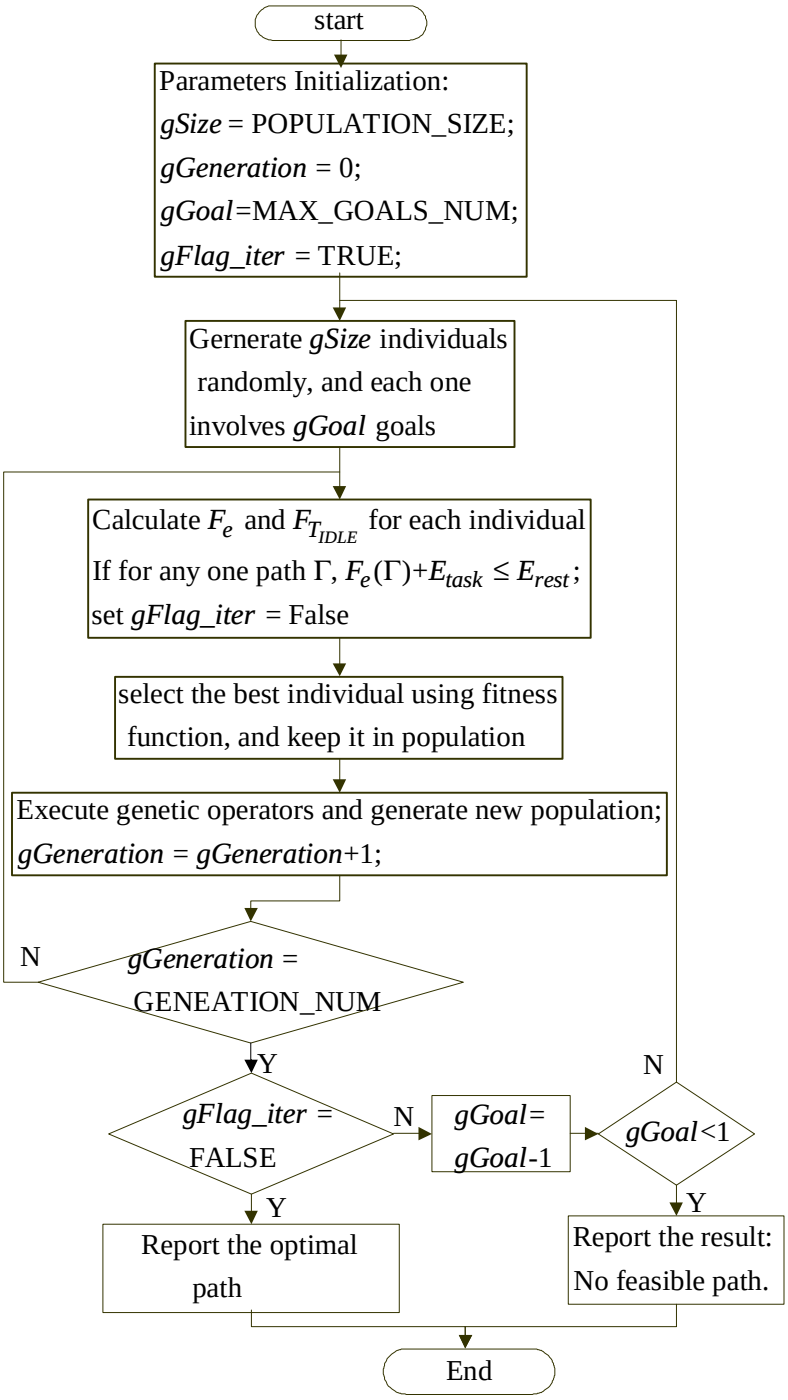

Figure 6. Execution of the proposed genetic algorithm

Several parameters should be emphasized. *gSize* is the population size, and *gGeneration* is the number of

generations. $gGoal$ indicates how many goals are involved in each individual, and $gFlag\_iter$ is used for determining if another circulation of performing the basic genetic algorithm is needed. In the entire procedure, $gSize$ keeps being $POPULATION\_SIZE$ that is constant. On the contrary, $gGoal$ may reduce if $gFlag\_iter$ remains $TRUE$ when beginning another cycle. At the beginning, the value of $gGoal$ is MAX_GOALS_NUM which is the total number of goals to be visited. If $gFlag\_iter$ remains $TRUE$ when one cycle is over, $gGoal$ may reduce by one until it decreases to zero, which means trying to find an optimal path with one less goal. If in one cycle, $gFlag\_iter$ turns into $FALSE$, the whole procedure will finish when $gGeneration$ increases to $GENERATION\_NUM$ which is a predefined constant.

In each cycle, the basic tailored genetic algorithm is conducted to search the optimal path of corresponding number of goals. In the end, if $gGoal$ is not smaller than one, then one optimal path is gotten, otherwise, it will report that no feasible path is available. This mechanism will guarantee that the algorithm obtains an optimal path involving as more goals as possible if at least one feasible path exists.

4. Simulation Studies

In this section, simulations are implemented to examine our proposed tailored genetic algorithm. On top of this, with the simulation results, analysis and discussion are addressed in detail.

4.1 Simulations and Results

We still use the topological map (noted as C_l) shown in Fig. 7 in simulations, which is built in previous work [1]. In addition, the attributes of each segment are also listed in that literature.

In simulations, parameters in the proposed genetic algorithm are set as follows: $POPULATION\_SIZE$=30, and $GENERATION\_NUM$=100. Crossover rate is $P_c = 0.9$ and Mutation rate $P_m = 0.001$. To simplify the computation, we assume $e_{task} = \tau = 0$, then $E_{task} = 0$. Therefore, equation (8) becomes $F_e(\Gamma) \le E_{rest}$.

Furthermore, we assume that E_{rest} is a constant. In the following simulations, we set $E_{rest} = 0.17V$.

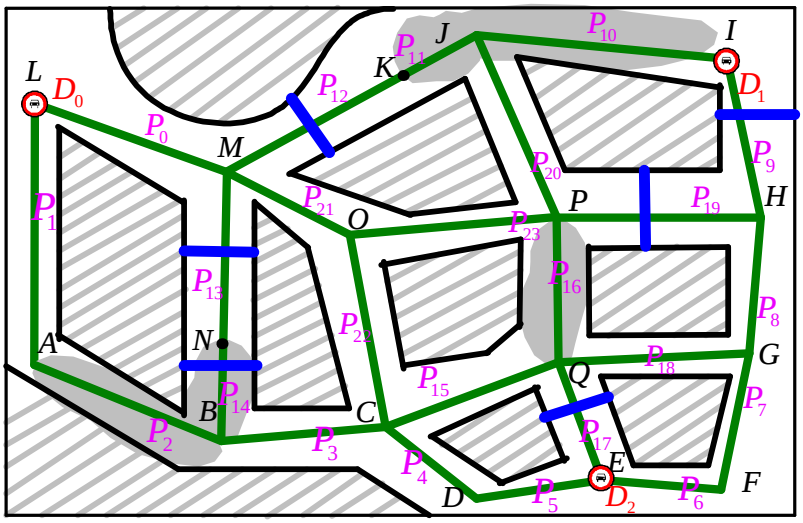

Figure 7. Topological map of environment.

(1) Simulation I

In this test, node A is set as the start point, and the goals are C, H and M. The energy consumption and idle time of the best individual in each generation are shown in Fig. 8. It is obtained from the result that the optimal solution comes out in the 18th generation. The optimal path is $A-B-C-O-M-O-P-H$. Its energy consumption $F_e = 0.1511V$ and idle time $F_{T_{IDLE}} = 1022.1330s$. Since $F_e < E_{rest} = 0.17V$, this path is feasible, and all goals can be visited. Further more, the order of visiting is C, M, H.

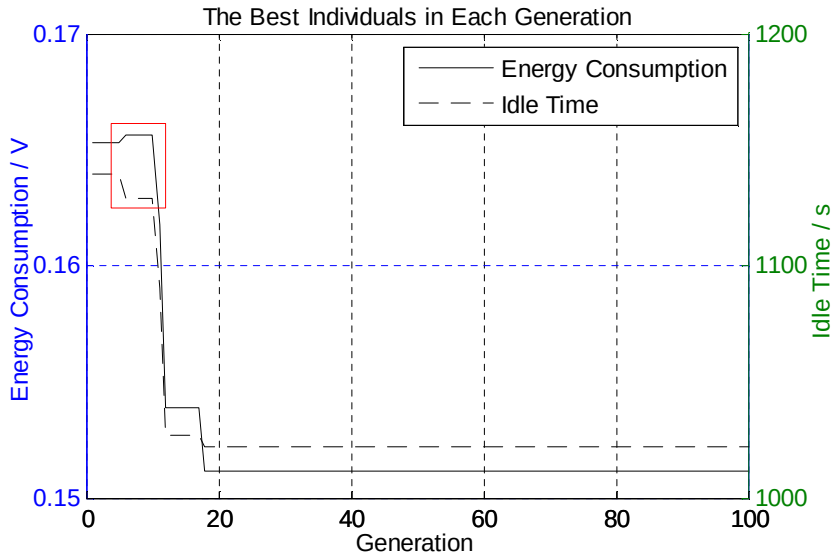

Figure 8. Result of simulation I.

In addition, there is a notable situation shown in the red rectangle in Fig. 8. For the best individual found in the 6^{th} generation, its idle time is less than that of the 5^{th} generation, while the energy consumption is more. This is because in the former 5 generations, we have found one path satisfying the condition of energy constraint, then the minimal idle time will be employed as principal for selecting the optimal path.

Table 1.
Details of best individuals in each generation.

Generation	Best individual	F_e / V	$F_{T_{IDLE}} / s$
1-5	$A-B-C-B-N-M-O-P-H$	0.1653	1139.6626
6-10	$A-L-M-N-B-C-Q-G-H$	0.1656	1128.5902
11	$A-L-M-O-C-O-P-H$	0.1617	1090.5179
12-17	$A-L-M-O-C-Q-G-H$	0.1540	1026.6969
18-100	$A-B-C-O-M-O-P-H$	0.1511	1022.1323

For sake of further understanding, we list out the concrete data of energy consumption and idle time of the best one in each generation in table 1. In table 1, the energy consumption of the best one in the first generation is $0.1653V$ that is less than $0.17V$. So, from the second generation, the best individual is selected from the candidates that satisfying energy condition by using idle time as criterion.

(2) Simulation II

In this simulation, we set A as the start point, and the goals are H,N,O and Q. One result is shown in Fig. 9. In the first 29 generations, energy consumption keeps more than $0.17V$, thus, during this period, the criterion for selecting optimal individual in each generation is energy consumption. In the 30th generation, we get the optimal path that is $A-B-N-M-O-C-Q-G-H$. Its energy consumption is $0.1646V$ and idle time is $1123.5112s$. The result shows that the robot also has sufficient energy visiting all the goals, and the order is N, O, Q and H.

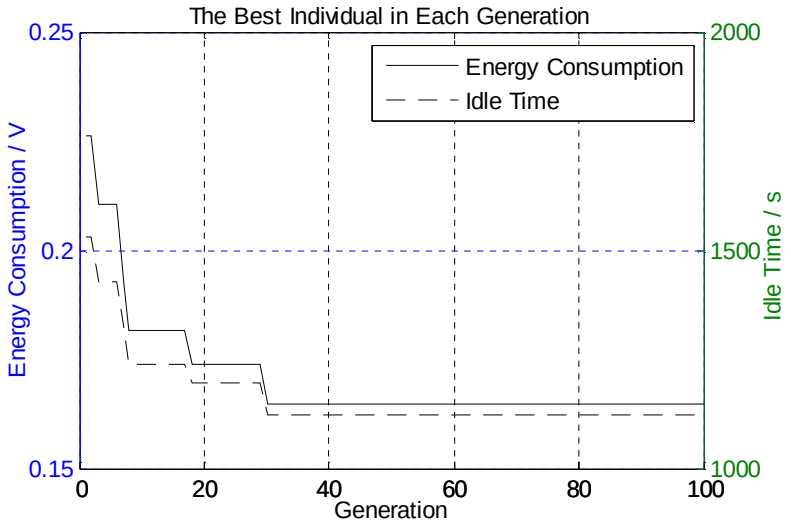

Figure 9. Result of simulation II.

(3) Simulation III

In this simulation, A is the start point, and goals are E, H, N and O. We show the result in Fig. 10.

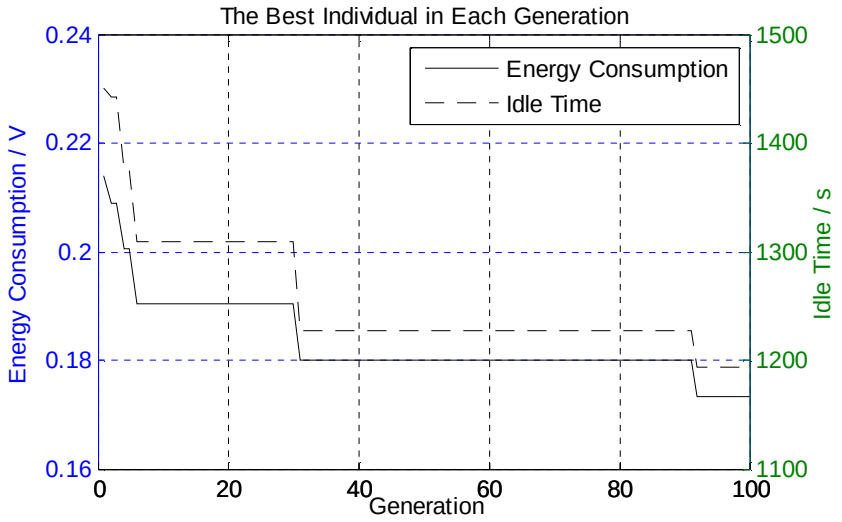

Figure 10. Result of simulation III.

From Fig. 10, we get the optimal solution in the 92nd generation in the first cycle. The energy consumption F_e is $0.1734V$ and idle time is $1193.8413s$. Because F_e exceeds the remaining available energy $0.17V$, this path is not feasible and we will continue searching another optimal path with one goal less. With this strategy, a new optimal path that both satisfies requirement of energy consumption and has minimum idle time is generated: $A\text{-}B\text{-}N\text{-}M\text{-}O\text{-}P\text{-}H$. Its energy consumption is $0.1274V$ and idle time is $886.9106s$.

The energy consumption is less than $0.17V$, however, it just allow the robot to visit three goals which are N, O and H.

4.2 Analysis of the simulation results

In the three simulations above, we implement our proposed tailored genetic algorithm to find the optimal path for multi-goal visiting task and finally optimal solutions are obtained. In the following we will discuss about the similarity and difference between each case and evaluate the proposed genetic algorithm based on simulation results.

(1) As the genetic algorithm itself is a kind of stochastic, evolutionary search method, the optimal solution obtained at the end may not be the global optimal one truly, but converges to.

(2) In the three cases above, the speed of converging to the optimal solution is different. For example, the optimal one appears in the 18th generation in simulation I, while it is obtained in the 30^{th} generation in simulation II. Moreover, in simulation III, in the first cycle, the optimal one is gotten in the 92^{nd} generation, and is proven to be not feasible.

(3) Generally, when using GA method, the stop condition can be either that the best solution keeps unvaried for certain number of generations, or that the current maximum generation is exceeded [40]. In proposed genetic algorithm, the latter is adopted. However, in reality, both can not ensure the final solution is one hundred percent the optimal one, and therefore it is uncertain that which one is better absolutely. As an example, in simulation III, the solution generated firstly in the 30^{th} remains the best one in the following 62 generations. If we use the former stop criterion, and set the maximum generation is 50 or 60, this solution will be regarded as the final optimal path. However, it is soon replaced by a better solution. Furthermore, if we set the maximum generation is 90, we also can not get the better solution that comes out soon.

(4) In all the simulations, parameter E_{rest} is considered as a constant for simplifying the problem.

Whereas, since for different paths, the last goal reached may not be the same, this value can be different. Actually, this value should be the minimum energy the robot needs to move to one charging station from the last goal. Hence, in practice, it is wiser to use different values of E_{rest} rather than the same fixed value.

(5) In the topological map, there have just 17 path nodes. Since it is not a big number, the advantage of our method cannot be reflected on time complexity. In relevant studies on analogous problems, generally there are two kinds of solutions, which are stochastic search algorithm and exhaustive search method. The second method can guarantee that the optimal solution is sure to be found. However, when searching space grows exponentially as the nodes increases, time complexity will grow enormously too. Even in extreme cases, it seems impossible to complete in acceptable time. Under this circumstance, stochastic evolutionary search such as the proposed genetic algorithm great advantage because it can quickly locate high performance regions in extremely large and complex search space [10].

Conclusion

We have proposed a tailored genetic algorithm to plan an optimal path for the multi-goal visiting task. Aiming at the particularity of the problem, special form of chromosome is used to represent the path and customized genetic operators are development. The effectiveness of the method is verified by simulations. Furthermore, through analysis of simulation results, evaluation on our proposed method is addressed, which is useful for wider implementation in various circumstances. Future work will be carried out to perfect this method by considering parameters more realistically and to test it on the real systems.

Acknowledgements

This research was sponsored by the Key Project of Science and Technology Committee of Chongqing (CSTC, 2009AB2139).

References

[1] F. Liu, S. Liang, and X. D. Xian, Determination of An Optimal Return-path on Road Attributes for Mobile Robot Recharging, *International Journal of Advanced Robotic Systems*, 8(5) , 2011, 83-92.

[2] F. Liu, S. Liang, and X. D. Xian, Corrigendum to Determination of An Optimal Return-path on Road Attributes for Mobile Robot Recharging, *International Journal of Advanced Robotic Systems*, 9(5) , 2012, 1-3.

[3] K. Z. Wang, S. Liang, H. B. Bi, and X. D. Xian, Implementation of a robot inspection system for substation equipment based on pioneer 3-at, Science, 4(5), 2010, 1-6.

[4] O. Madsen, C. B. Sørensen, R. Larsen, L. Overgaard, and N. J. Jacobsen, A system for complex robotic welding, Industrial Robot: An International Journal, 29(2), 2002, 127-131.

[5] R. Thrapp, C. Westbrook, and D. Subramanian, Robust localization algorithms for an autonomous campus tour guide, *Proceedings of IEEE International Conference on Robotics and Automation*, 2001, 2065-2071.

[6] P. Raja and S. Pugazhenthi, Optimal Path Planning of Mobile Robots: A Review, *International Journal of Physical Sciences*, 7(9), 2012, 1314-1320.

[7] N. Sariff and N. Buniyamin, An Overview of Autonomous Mobile Robot Path Planning Algorithms, *4th Student Conference on Research and Development*, 2006, 183-188.

[8] M. Yuan, S. A. Wang, C. Wu, and N. Chen, A Novel Immune Network Strategy for Robot Path Planning in Complicated Environments, *Journal of Intelligent & Robotic Systems*, 60(1), 2010, 111-131.

[9] M. McNaughton and C. Urmson, FAHR: Focused A* Heuristic Recomputation, *International Conference on Intelligent Robots and Systems*, 2009, 4893-4898.

[10] T. Shibata and T. Fukuda, Intelligent Motion Planning by Genetic Algorithm with Fuzzy Critic, *International Symposium on Intelligent Control*, 1993, 5-10.

[11] O. Wongwirat and A. Anuntachai, Searching Energy-Efficient Route for Mobile Robot with Ant Algorithm, *11th International Conference on Control, Automation and Systems*, 2011, 1071-1075.

[12] S. X. Yang and Y. Hu, A Knowledge Based GA for Path Planning of Multiple Mobile Robots in Dynamic Environments, *IEEE International Conference on Robotics and Automation*, 2007, 71-76.

[13] M. Bennewitz and S. Thrun, Optimizing Schedules for Prioritized Path Planning of Multi-Robot Systems, *IEEE International Conference on Robotics and Automation*, 2001, 271-276.

[14] M. Mansouri, M. A. Shoorehdeli, and M. Teshnehlab, Path Planning of Mobile Robot Using Integer GA with Considering Terrain Conditions, *IEEE International Conference on Systems, Man and Cybernetics*, 2008, 208-213.

[15] T. Zheng and P. Wang, Priority based Dynamic Multiple Robot Path Planning, *2nd International Conference on Autonomous Robots and Agents*, 2004, 373-378.

[16] L. E. Parker, Path Planning and Motion Coordination in Multiple Mobile Robot Teams, *Encyclopedia of Complexity and System Science*, 2009, 1-24.

[17] T. Hellstrom and O. Ringdahl, Real-time path planning using a simulator-in-the-loop, I*nternational Journal of Vehicle Autonomous Systems*, 7(1/2), 2009, 56-72.

[18] R. Iraji, Robot path planning using wavefront approach with wall-following, *2nd IEEE International Conference on Computer Science and Information Technology*, IEEE, 2009, 417-420.

[19] H. F. Wang. and Y. Z. Yang, Time-optimal Trajectories for a Car-like Robot, *Automatica*, 34(4), 2008, 445-452.

[20] Z. Sun and J. H. Reif, On Finding Energy-Minimizing Paths on Terrains, *IEEE Transaction on Robotics*, 21(1), 2005, 102-114.

[21] S. Liu and D. Sun, Optimal Motion Planning of a Mobile Robot with Minimum Energy Consumption, *International Conference on Advanced Intelligent Mechatronics (AIM2011)*, Budapest, Hungary, 2011, 43-48.

[22] Y. Mei, Y.-hsiang Lu, Y. C. Hu, and C. S.G. Lee, Energy-Efficient Motion Planning for Mobile Robots, *International Conference on Robotics and Automation*, 2004, 4344–4349.

[23] E. Lobaton, J. Zhang, S. Patil, and R. Alterovitz, Planning Curvature-Constrained Paths to Multiple Goals Using Circle Sampling, *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, 1463-1469.

[24] F. Liu, S. Liang, X. D. Xian, and H. B. Bi, Optimal Path Planning for Mobile Robot in Consideration of Road Attributes, *ICIC Express Letters*, 6(1), 2012, 281-287.

[25] N. Sedaghat, Mobile Robot Path Planning by New Structured Multi-objective Genetic Algorithm, *The 3rd International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2011, 79-83.

[26] F.O. Castillo, L. Trujillo and P. Melin, Multiple objective genetic algorithms for path-planning optimization, *Soft Computing*, 11, 2007, 269-279.

[27] S. X. Yang, Y. Hu, and M. Q. Meng, A Knowledge Based GA for Path Planning of Multiple Mobile Robots in Dynamic Environments, *IEEE International Conference on Robotics and Automation*, 2007, 71-76.

[28] G. Capi, Multiobjective Evolution of Neural Controllers and Task Complexity, *IEEE Transactions on Robotics*, 23(6), 2007, 1225-1234.

[29] E. Masehian and D. Sedighizadeh, A Multi-Objective PSO-based Algorithm for Robot Path Planning, *IEEE International Conference on Industrial Technology(ICIT)*, 2010, 465-470.

[30] R. T. Marler and J. S. Arora, Survey of multi-objective optimization methods for engineering, *Struct Multidisc Optim*, 26, 2004, 369-395.

[31] G. Avigad and K. Deb, The Sequential Optimization-Constraint Multi-objective Problem and its Applications for robust Planning of robot paths, *IEEE Congress on Evolutionary Computation*, 2007, 2101-2108.

[32] Holland, Adaptation in Natural and Artificial Systems, *Ann Arbor: University of Michigan Press*,

1975.

[33] A. Konak, D. W. Coit, and A.E. Smith, Multi-objective optimization using genetic algorithms: A tutorial, *Reliability Engineering*, 91(9), 2006, 992-1007.

[34] I. AL-Taharwa, A. Sheta, and M. Al-Weshah, A Mobile Robot Path Planning Using Genetic Algorithm in Static Environment, *Journal of Computer Science*, 4(4), 2008, 341-344.

[35] G. Nagib and W. Gharieb, Path Planning for a Mobile Robot Using genetic Algorithms, *Proceedings of the International Conference on Electrical, Electronic and Computer Engineering (ICEEC'04), Cairo, Egypt*, 2004, 185-189.

[36] J. Tu and S. X. Yangt, Genetic Algorithm Based Path Planning for a Mobile Robot, *International Conference on Robotics and Automation*, 2003, 1221-1226.

[37] M. Mansouri, M. A. Shoorehdeli, and M. Teshnehlab, Integer GA for Mobile Robot Path Planning with using another GA as repairing function, *Proceedings of the IEEE International Conference on Automation and Logistics*, 2008, 135-140.

[38] Z. Yao, L. Ma, A Static Environment-Based Path Planning Method by Using Genetic Algorithm, *International Conference on Computing, Control and Industrial Engineering*, 2010, 405-407.

[39] F. Ahmed and K. Deb, Multi-Objective Path Planning using Spline Representation, *IEEE International Conference on Robotics and Biomimetics (ROBIO)* , 2011, 1047-1052.

[40] E. Zitzler, M. Laumanns, and S. Bleuler, A Tutorial on Evolutionary Multiobjective Optimization, *Evolutionary Computation*, 535(5), 2004, 3-37.