A Genetic Algorithm for Path Generation and its Applications

Jun Inagaki*, Tomoaki Shirakawa†, Tetsuo Shimono† and Miki Haseyama‡

* Hokkaido Institute of Technology, Sapporo 006-8585 Hokkaido Japan

E-mail: inagaki@hit.ac.jp Tel: +81-11-681-2161

† Tokai University, Sapporo 005-8601 Hokkaido Japan

E-mail: shira@tspirit.tokai-u.jp, shimono@tspirit.tokai-u.jp Tel: +81-11-571-5111

‡ Graduate School of Information Science and Technology, Hokkaido University, Sapporo 060-0814 Hokkaido Japan

E-mail: miki@ist.hokudai.ac.jp Tel: +81-11-706-6077

Abstract—**Path generation is an optimization problem mainly performed on grid maps that combines generation of paths with minimization of their cost. Several methods that belong to the class of exhaustive searches are available; however, these methods are only able to obtain a single path as a solution for each iteration of the search. Conversely, while genetic algorithms involving a type of multipoint search methods have been proposed as suitable candidates for this problem with the goal of simultaneously searching for multiple candidate paths, these methods are limited to particular applications, and there are limitations on the types of paths that can be represented. This paper therefore proposes a path generation method that is applicable to more general-purpose applications compared to previous methods based on a new design of the genotype used in the genetic algorithm.**

I. INTRODUCTION

The shortest route problem involves searching for the minimal-cost route joining specified starting and ending points, and it is mainly solved using exhaustive search methods such as the depth-first search (DFS)[1], breadth-first search (BFS)[2], and Dijkstra's algorithm[3]. The authors have also proposed a variety of routing methods[5], [6] that use genetic algorithms[7].

Numerous fields, including the design of new roads and the creation of shipping lanes, involve processed that, in addition to being able to simply search for paths on a graph, should be able to generate paths that minimize the total cost accumulated, by traversing the segments in a search space on which some predefined cost distribution has been configured (e.g., a mesh or grid map; referred to as "path generation" in this paper), and there is thought to be a strong demand for search methods having high efficiency.

Besides simply searching for routes on a graph, there is thought to be a strong demand for generating paths that minimize the total cost accumulated, by traversing the segments in a search space on which some predefined cost distribution has been configured. Path generation can be applied to numerous fields, including the design of new roads and the creation of shipping lanes.

Although the optimal path can be obtained using exhaustive search methods when performing path generation on a grid square map, the search space is significantly larger compared to routing on a graph, and the need for a large number of calculation poses a problem. Furthermore, for algorithms based on exhaustive search methods that only find the optimal solution, it is difficult to generate multiple candidate paths that have small costs in a single iteration.

The present research therefore considers an algorithm that can obtain a set of multiple paths with small costs from a single iteration search process using a genetic algorithm. Genetic algorithms have the property of producing many possible solutions during the course of the search process. The authors have previously proposed a procedure of searching for multiple routes in a graph by exploiting this property. The goal of the present research is to simultaneously generate multiple candidate paths by utilizing this same property through a suitable genetic algorithm process.

Although Ref. [8] offers one example of path generation using a genetic algorithm, it is limited to generating flight paths for aircraft, and is therefore unable to create paths that deviate to a large extent from a straight line segment joining the specified starting and ending points. However, considering that the goal of this research is to apply path generation to general-purpose applications, it is necessary to be able to generate paths that, for example, spiral around the starting or ending point.

This paper therefore proposes a new method for generating more general-purpose paths. In the proposed method, paths that could not be represented by the method in Ref. [8] are made possible by designing a genotype with a fixed length and redundancy. Furthermore, deterioration of the search efficiency due to sudden reductions in the diversity of the population of chromosomes is prevented by using this genotype to suppress the generation of lethal chromosomes during crossover. Also, the effectiveness of the proposed method is checked by some experiments.

II. GENETIC OPERATORS IN THE PROPOSED METHOD

A. Genotype design

In the proposed method, the genotype is designed such that it is able to perform general-purpose path generation by improving on the genotype proposed in Ref. [8].
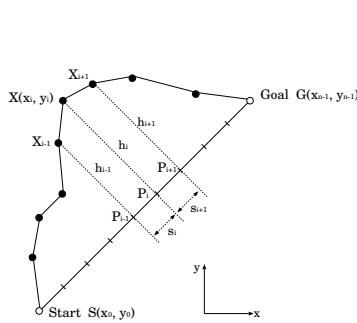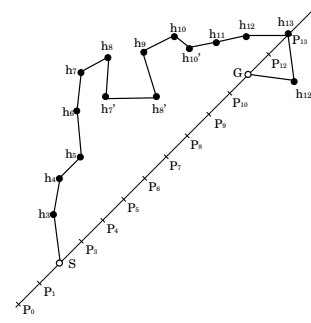
Fig. 1. Path model (earlier method).

Fig. 2. Path model (proposed method).

P_0	P_1	S	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	G	P_{12}	P_{13}
*	*	*	*	*	*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	h_7'	h_8'	*	h_{10}'	0	h_{12}'	*
*	*	0	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}	h_{12}	h_{13}

Fig. 3. Genotype related to the positional representation of the path.

P_0	P_1	S	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	G	P_{12}	P_{13}
1	1	1	1	1	1	1	1	1	1	1	0	-1	-1
1	1	-1	-1	1	2	-1	-1	1	2	-1	1	2	2
2	-1	1	2	-1	1	1	1	1	-1	1	0	-1	-1
1	-1	0	1	1	1	1	1	-1	1	2	1	1	-1

Fig. 4. Genotype related to the directional representation of the path.

We begin by explaining the path representation from Ref. [8]. Figure 1 shows the model of the paths. Let S and G be the starting and ending points, respectively. The line segment SG can then be divided into equal parts, and we let each of the points be $P_i(i = 1, \ldots, n)$ and the intervals between points be s_i. Further, if we consider X_i (x_i, y_i) to be the point at a distance h_i from P_i, then the path can be represented by a polygonal line from S to G via these X_i. Using this method enables the representation of relatively detailed paths, within the limitations posed by the available computational and memory resources, by reducing s_i. These paths are then represented by storing the distance h_i from P_i to X_i in the ith locus in a one-dimensional array of loci in the genotype.

However, because the goal of the method in Ref. [8] was to generate flight paths, it is unable to represent paths that spiral in to reach the starting or ending point similar to that shown in Fig. 2. That is, paths where there exists a point P_i, which corresponds to multiple points when projecting the path onto SG, cannot be represented by this method. This poses a problem when performing general-purpose path generation.

Let us suppose that it is possible to represent a path such as that shown in Fig. 2 using the genotype from Ref. [8]. The gene length would then vary among chromosomes, and we expect many lethal chromosomes created by performing crossover, causing a sudden drop in the diversity of the population of chromosomes after crossover. The same type of phenomenon has been demonstrated when genetic algorithms were applied to routing; in Ref. [4], this problem was avoided by using fixed-length genotype design. For the above reasons, the proposed method uses a fixed-length genotype design, as discussed below.

Figure 3 shows an example of the type of array that can be provided to represent the path shown in Fig. 2. In each gene, the distance h_i of the path from the line SG is stored sequentially from the bottommost row at the locus represented by P_i and a randomly generated value of h_i is stored in genes that do not contribute to representation of the path (represented by a * in the diagram).

Furthermore, an additional array like the example shown in Fig. 4 is provided for representing the direction of the path. For a point on the path represented by a gene in the array in Fig. 3, this array indicates the neighboring points to which this point connects to next. More specifically, the gene "1" represents a connection to the neighboring point on the path associated with P_{i+1} (in the genotype, this is the lowermost gene in the column on the right where the point has not yet been traversed), "−1" represents connection to the neighboring point on the path associated with P_{i-1} (in the genotype, this is the lowermost gene in the column on the left where the point has not yet been traversed), and "2" represents connection to the neighboring point on the path associated with P_i (in the genotype, this is the gene in the row above). The ending point is represented by "0". For example, the gene in Fig.4 that corresponds to the gene for h_8 in Fig. 3 is "−1", which indicates that this point connects to a point that has not yet been traversed from among the genes in the column on the left, i.e., h_7' (since the path in Fig. 2 is $\cdots \rightarrow h_7 \rightarrow h_8 \rightarrow h_7' \rightarrow \ldots$, the point h_7 has already been traversed).

The genes "1", "−1", and "2" are stored randomly in loci that do not contribute to representing the path (the elements that are not shaded in Fig. 4). If left in this state, however, it is possible for the chromosome to include a lethal gene after crossover, similar to that shown in Fig. 5, where the path does not connect the starting point S to the ending point G. The proposed genotype therefore contains "0" in the uppermost row of the ending point, "1" in the uppermost row of the genes to the left of the ending point, "−1" in the uppermost row of the genes to the right of ending point, either "1" or "2" in the leftmost column, and either "−1" or "2" in the rightmost column, as shown in Fig. 4. This is able to prevent
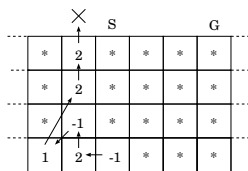
Fig. 5. Example of lethal gene.

lethal genes from being generated during crossover.

The genotype design described above is therefore made up of genes that contribute to representing the path and genes that do not, and there may exist different genotypes that have the same phenotype. The genotype therefore contains redundancy, because there is a one-to-many relationship between phenotypes and genotypes. Although non-redundancy has been recommended (that is, a one-to-one correspondence between chromosomes and candidate solutions), in the proposed method, the generation of lethal chromosomes that do not correctly represent a path after crossover is prevented by using a genotype that introduces redundancy. This phenomenon is caused by the same reason in the case of routing on a graph, as shown in Ref. [4].

Incidentally, increasing the number of columns in the genotype is equivalent to increasing the number of h_i, making possible to represent more detailed paths. Further, increasing the number of rows in the genotypes is equivalent to increasing the number of times that the path can pass through P_i when the path is projected onto the line segment SG, making it possible to represent paths that have higher degree of freedom. However, increasing the number of rows and columns is accompanied by an increase in the computational resources required for processing, and the configuration therefore needs to be selected by taking this trade-off into account.

B. Evaluation

The experiments in this paper involve path generation on a mesh map. Fitness is therefore calculated in the following manner. First, the polygonal line paths represented by the chromosomes are projected onto the mesh map (Fig. 6). Let the cells that the path passed through be g_i $(i = 0, \ldots, N)$ and the cost associated with each g_i be w_i. The evaluation function is then be expressed by the following equation.

$$f = \frac{1}{\displaystyle\sum_{i=0}^{N} w_i} \tag{1}$$

C. Generation of the initial population of chromosomes

The phenotypes are first created by randomly generating polygonal line paths similar to the one shown in Fig. 2. Note that these are created such that they satisfy the constraints imposed by the preconfigured number of rows in the genotype (the number of times the path can traverse the same P_i when projected onto the SG line segment). Next, the generated paths are coded into genotypes, and using the method shown in II-A, genes are then stored in the loci that do not contribute to the representation of the path.
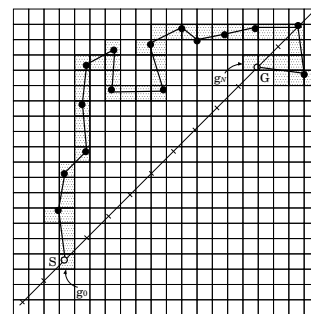

Fig. 6. Example of the path of a chromosome projected on to a mesh map.

D. Reproduction

Reproduction employs a roulette wheel selection which is conducted by spinning a biased roulette wheel sized in proportion to the fitness of each chromosome, and an elitist replacement whereby the best fitness chromosome is copied into the next generation[7].

E. Crossover

One-point crossover is used for the crossover. In the proposed method, a crossover point is selected randomly for two parent chromosomes that have been selected by reproduction, and all of the genes to the right of the crossover point are swapped between the two chromosomes. In the event that chromosomes, after crossover, form a chromosome that does not represent a path, the crossover point is reselected and crossover is performed again. A rare situation may arise where the chromosomes after crossover are lethal chromosomes for every crossover point, and in this case crossover is not performed.

F. Mutation

For loci that are selected according to the mutation probability, the value of the two genes (the positional and directional representations) are changed randomly if these genes do not contribute to the representation of the path. If the genes do contribute to the representation of the path, the path from the position represented by those genes to the end point is changed randomly.

III. EXPERIMENTAL RESULTS

This section describes experiments performed by applying a mesh-shaped cost map in order to verify the effectiveness of path generation using the proposed method. A 256×256 mesh map was prepared with costs in the range of 0 to 255 assigned to each of the cells (in Fig.7, dark cells indicate regions of higher cost). In the experiments, the line between the starting point S and ending point G was divided into 15 equal segments of length s_i and P_i was placed along SG and the straight line that extends SG.

The number of columns in the genotype (gene length) was set equal to the number of P_i, and the number of rows was set to 3. The genetic parameters were set as follows: the population size was set to 1000, the crossover rate was set to
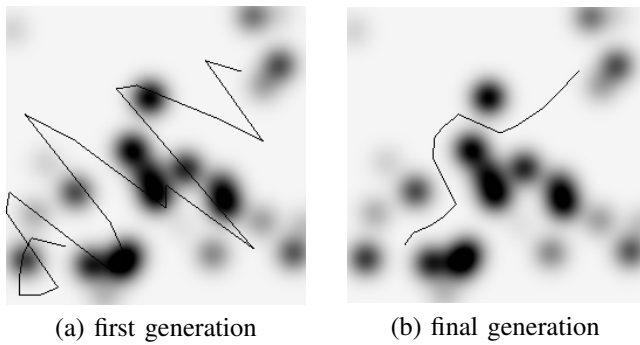
(a) first generation   (b) final generation

Fig. 7. Example of chromosome (weighted cost map)

Fig. 9. Path cost with generation change.

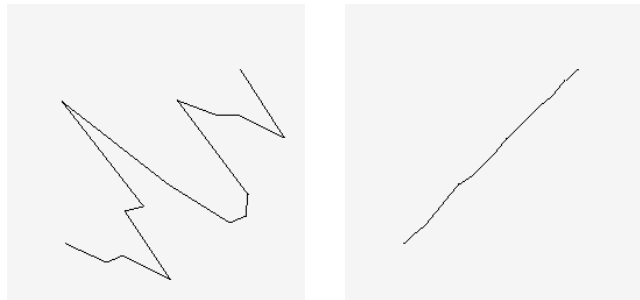

(a) first generation   (b) final generation

Fig. 8. Example of choromosome (unweighted cost map)

1.0, the mutation probability was set to 0.01, and the number of generation was set to 1000.

We now present the results of applying the proposed method using the above settings. Figure 7(a) shows an example path from the initial population of chromosomes. From the figure, it is clear that this method is able to represent paths that rotate around the starting point, which was not possible in the previous method. The best chromosome after 1000 generations is shown in Fig. 7(b). It is clear that compared to the initial generation, a path with a shorter length and which avoid high cost regions was obtained.

Figure 8 shows a path in the initial generation and a path resulting from path generation for the case where a cost map was applied with the same cost everywhere, respectively. By comparing the two figures, it is clear that the population has evolved, and that the path represented by the chromosome is close to the optimal solution (which is the straight line segment SG in this experiment). Figure 9, which shows the path length of the best chromosome and the average path length of all chromosomes at each generation in a given execution of the algorithm, demonstrates that as the generations passed, chromosomes with shorter path length were generated by crossover, and that was a trend for the average path length of the population to decrease due to increase in the number of these chromosomes by reproduction.

IV. CONCLUSION

A path generation method using a genetic algorithm was proposed in this paper.

The ordinary methods based on exhaustive search have a tendency to require a heavy calculation cost when applying
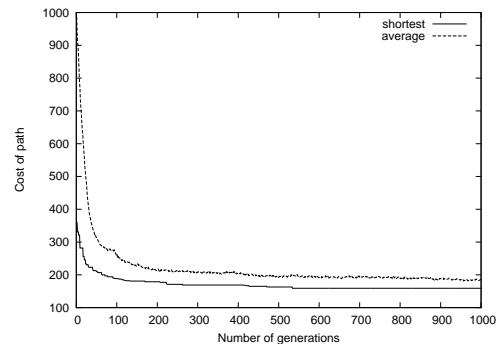
them to large maps. On the other hand, the proposed method can reduce the calculation cost because it does not depend on the map size. Moreover, genetic algorithms generate multiple possible solutions in a search process, though the exhaustive search methods search for the optimum only. Hence, in the future, the proposed method will be able to obtain a set of multiple paths with small costs in a single iteration search process similarly to [4] and [8].

The previous GA method[8] was unable to create paths that deviate to a large extent from a straight line segment joining the specified starting point and ending point, because a one-dimensional array is used for its genotype. However, the new genotype of the proposed method represented the paths using two-dimensional fixed-length arrays that combined an array representing the positions of the path with an array representing the path's direction of advance. Furthermore, by designing genetic operators suitable for this genotype, this method could represent complicated paths that could not be handled by the previous method, thus making it possible to use a genetic algorithm for a more general-purpose path generation application.

Moreover, in the future, candidate paths beside the optimum path will be obtained by utilizing the property of genetic algorithms which generate many possible solutions. For example, using the fitness values with giving weight to each selected region on the map is one of the ideas for this kind of searching.

REFERENCES

[1] E. F. Moore, "The shortest path through a maze," Proc. Int. Symp. Switching Theory, part II, pp. 285–292, 1959.
[2] R. E. Tarjan, "Depth-first search and linear graph algorithms," SIAM J. Comput., vol. 1, pp. 146–160, 1972.
[3] E.W. Dijkstra, "A note of two problems in connection with graphs," Numerische Mathematik, vol. 1, pp. 269–271, 1959.
[4] J. Inagaki, M. Haseyama, and H. Kitajima, "A method of determining various solutions for routing application with a genetic algorithm," IEICE Trans. Inf. & Syst., vol. J82-D-I, no. 8, pp. 1102–1111, 1999.
[5] J. Inagaki, M. Haseyama, and H. Kitajima, "A new fitness function of a genetic algorithm for routing applications," IEICE Trans. Inf. & Syst., vol. E84-D, no. 2, pp. 277–280, 2001.
[6] J. Inagaki, M. Haseyama, and H. Kitajima, "A genetic algorithm for routing with an upper bound constraint," IEICE Trans. Inf. & Syst., vol. E88-D, no. 3, pp. 679–681, 2005.
[7] D.E. Goldberg, Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, 1989.
[8] M. Tanaka, M. Mizoguchi, T. Tange, I. Takami, and S. Suzuki, "Multiroute planning for aircraft using genetic algorithms," Systems and Computers in Japan, vol. 35, no. 2, pp. 39–48, 2004.