GA-based Path Planning for Mobile Robots: An Empirical Evaluation of Seven Techniques

Alison Watkins, Ph.D.
College of Business, University of South Florida St. Petersburg, FL 33730
Email: awatkins@usfsp.edu

***Abstract*—Previous research suggests that genetic algorithms (GAs) offer a promising solution to path planning for mobile robots. We examine six simple GAs used in prior studies, comparing them to a new node sequence approach that includes a two-step fitness function. Through a series of repeated trials using a simple 16x16 grid, a 100x100 grid, a 600x600 Mars landscape, and a complex maze-like environment, we compare the chromosome structures and fitness functions of these seven methods. The results of our empirical testing indicate that the proposed dual goal approach, which uses a fixed length chromosome structure, outperformed both monotonic and other node sequence approaches, consistently finding a feasible path in even the most challenging of these environments.**

***Index Terms*—artificial intelligence, routing, genetic algorithms, robotics.**

I. Introduction

Autonomous mobile robots, which have the ability to navigate without direct human control, have potential value in a wide variety of settings, including factories, warehouses, mining, construction, healthcare, and space exploration. To operate in natural settings such as these, however, robots must be able to map the environment, and plan their path of movement to minimize distance, time, and other performance objectives, while avoiding obstacles [1]. Thus, path planning is an area of research that has garnered considerable attention in recent years.

Finding an optimal path is typically a computationally intensive task that is necessary to goal achievement, but does not fulfill the robot's primary goal. Because the robot's processing capabilities are limited, researchers have sought to develop efficient path planning algorithms that conserve resources and quickly find a clear path. Evolutionary search techniques such as genetic algorithms (GAs) are especially well-suited to this task because they are robust, relatively easy to apply, and capable of finding the global optimum via a search of the entire space [2]. Key to their success are two design issues -- the construction of the chromosomes, which capture path information, and the specification of the fitness function used to evaluate them.

Prior research has described successful approaches to the path planning problem [3] and a number of GA-based approaches to path planning under various scenarios; however, their relative effectiveness has not been empirically assessed. As a result, no guidance or recommendation as to the most efficient GA-Based approach has emerged. To establish a baseline for assessing more complex techniques and scenarios, we evaluate seven GA-based planning approaches that use a simple GA, working in 2D, for a single robot with a single goal. Through repeated trials using a 16x16 grid, a 100x100 grid, a 600x600 Mars landscape image, and a more complex maze-like environment, we conduct statistical comparisons, demonstrating that a fixed chromosome design coupled with a two-step fitness function that does not accept infeasible paths may be the most efficient and effective technique for path planning in this scenario.

II. Genetic algorithms for Path Planning

Genetic algorithms begin with a population of chromosomes representing random solutions to a particular search problem. Each chromosome or solution is first tested and the results are evaluated by a fitness function to determine how close it came to achieving the search objective. Based on the resulting fitness values, the chromosomes are then subjected to genetic operations to form a new population of potential solutions. In simple GAs, diversification of the chromosomes is achieved through a process of selection and recombination based on crossover and mutation. In more complex, hybrid GAs, other specialized operators may be added to assist in this process. As each new population is generated, the fitness values increase the likelihood that better solutions will survive and contribute to the evolving gene pool. These actions continue until a stopping condition is met -- either a set number of generations has been created and evaluated or an acceptable solution has been found. (For a more complete introduction to genetic algorithms, see [4,5]).

Like other adaptive search techniques, GAs are not guaranteed to find an optimal solution, but they have proven useful in finding very good solutions where time and system resources are constrained [5]. Because GAs are especially well-suited to problems where the search is

Manuscript received July 24, 2012; accepted November 19, 2012.
This work was supported in part by the Florida Space Grant Consortium under Grant No. 0000029716

doi:10.4304/jcp.8.8.1912-1922

large, complex, or not well understood, they are of interest in path planning, where they have been used to generate efficient routes between two or more locations [6 - 17].

The literature also provides many examples of path planners that use a GA in conjunction with other specialized optimization algorithms, for example, neural networks [18,19], synthetic pheromones [20], B-Spline Curves [21], A* Algorithm [22], rough sets [23] and fuzzy logic [24]. In addition, a number of researchers have extended the basic GA approach, producing the Immune GA [25] and the Hierarchical GA [26]. Although these extended approaches are potentially quite powerful, this research focuses on evaluating the simple GA for point-to-point 2D path planning.

The performance of a GA on a given search problem depends on a number of factors, including the size of the population in each generation, the genetic operators that are applied in the reproductive process and their values, the number of generations allowed, the construction of the chromosomes, and the fitness function used in evaluating them. Because many prior studies using GAs for path planning focus on different scenarios, varying two or more of these factors, it is very difficult to make comparisons across studies or draw conclusions as to the most effective techniques. We address the issue of non-comparability via a series of experiments designed to evaluate the relative performance of six previously-published GAs as well as one new one, focusing specifically on two issues – the construction of the chromosomes and the fitness functions that are used to assess them – while holding other GA features constant.

A. Chromosome Construction

The first step in formulating a GA to solve a particular search problem is translating the problem into biological terms by encoding information in the chromosomes. In path planning problems, each chromosome represents a series of straight line segments or a sequence of nodes that, taken together, describe a path. Any given chromosome or path can be *feasible*, if no obstacles are encountered, or *infeasible*, if any of the intermediate nodes contains an obstacle that would interfere with the robot's movement.

Techniques for encoding a path vary in terms of the actual genotype structure, i.e., the number of variables used to describe each movement. Generally, simpler genotype structures require less processing time, while more complex structures provide greater flexibility in terms of allowable moves. Additionally, some approaches utilize a fixed length chromosome, while others use a variable length chromosome. In many of the fixed length approaches, the rows (or columns) of the grid space are represented by the gene's position within the chromosome, while the column (or row) coordinates are stored within the genes. Thus, for an $n \times n$ grid, a fixed length chromosome would contain at a minimum, n genes. In variable length approaches, both the x and y coordinates are represented in some way in the individual genes, either explicitly or as numbered grid locations. Prior research suggests that fixed length representations are typically very efficient and, because the genes are specifically organized, very logical and easy to interpret [27]. On the other hand, variable length approaches may be more adaptable [27] and better suited to dynamic environments with many obstacles [12], but the traditional genetic operators must be modified to process them [28]. (Further issues associated with variable length chromosomes are discussed in later sections of this paper.)

B. Fitness Function

After each of the chromosomes is tested, a fitness function is used to evaluate it, determining how close it comes to satisfying the search goal. How quickly a GA converges on an acceptable solution depends in large measure on the objective function used in evaluating the performance of each chromosome. The fitness function in effect guides the search, based on the knowledge and experience of previous chromosomes, rewarding those that are "better" in some respect and punishing others with poorer fitness values [29]. Not only can a well-constructed fitness function improve the likelihood of finding a short, obstacle free path in fewer iterations but, when coupled with an efficiently-sized chromosome, it will consume fewer system resources in the process [5].

Previous research on path planning has utilized a wide range of measures to assess the fitness of candidate solutions, including the distance between moves, the length of the path, collisions with solid obstacles, the number of turns or the smoothness of the path, etc. However, without empirical assessment of these alternative approaches, no firm conclusion can be reached with respect to the best fitness measures for robot guidance. The experiments described in the remainder of this paper compare the chromosome structures and fitness functions described in six previously published path planning studies as well as one new approach. Although several of the original demonstrations used hybridized GAs that incorporated domain-specific operators, these operators were not included in our experiments to enable direct comparison of the chromosome structures and fitness functions, without the potentially confounding effects of such ad hoc efficiency improvements [30].

III. GA-Based Path Planning Approaches

A review of the literature identified six GA-based path planning approaches for single point mobile robots with a single goal as shown in Table I. In each of them, the tested path is allowed to travel through obstacles; however, they differ in terms of how such infeasible segments are evaluated by the fitness function. One new technique, the Dual Goal Approach was also included in these experiments. In the sections that follow, each approach is first described and its navigation abilities demonstrated using the 16x16 grid shown in Fig. 1, which contains 8 obstacles (shown as black squares). In this sample problem, the goal of each path planner is to develop an efficient route from the starting position is {1,1} (upper left corner) to the goal {16,16} (lower right corner) while avoiding all obstacles. Some of the

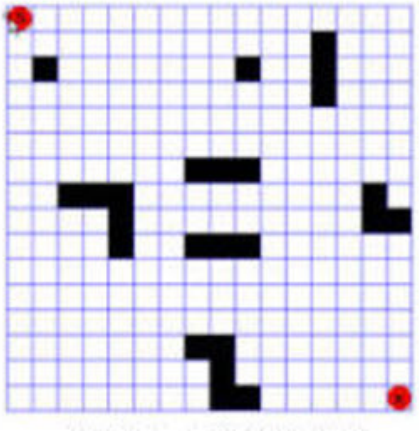


Figure 1. Demonstration Grid

approaches have an additional objective which is to reach the goal with the fewest number of turns. The shortest route through this grid is 30 moves with only two turns.

A. The Monotonic Approaches

The first two approaches listed in Table I can be defined as monotonic insofar as a path is represented by a row- or column-wise sequence of steps. For example, in row-wise movement the next move on the x coordinate is represented by the genome position and the y by the genome value. The advantage to this approach is that the length of the chromosome can be fixed, potentially speeding up real-time processing [28, 34, 35]. On the other hand, although GAs using this approach may be able to conquer most paths, they sometimes have difficulty navigating around obstacles and thus they can miss an optimum path [28].

Local Path Planner (LPP)

The local path planner, described by [35,36], was designed to extend earlier methods [34], using both row- and column-wise planning. The chromosome consists of four parts: The first value is a path-flag to indicate whether the path should travel in a row- or column-wise manner. The second part gives the *x,y* coordinate for the path. For example, in the chromosome fragment shown in Table II, position 2 indicates both the gene number and the *x* coordinate value, while 10 is the actual value of the gene and represents the y coordinate on the grid. Therefore, the first move from the starting position {1,1} would be to position {2,10}.

TABLE II
EXAMPLE OF PARTIAL CHROMOSOME FOR LOCAL PATH PLANNER (LPP)

Position	Value
2	10
3	8
4	3
5	0
6	9
7	11
8	14
9	14
10	11
11	1
12	2
13	5
14	10
15	8

The third part of the chromosome (not shown) is the path direction, which indicates whether travel should be initiated in a horizontal or vertical direction. The final two bits are described as path switch indicators, which indicate when a route should switch from a row- to column-wise direction. Each chromosome has the possibility of changing directions twice. The total length of the chromosome for a 16x16 grid would include 35 values.

LPP uses a relative fitness function where each chromosome is assessed based on its performance in relation to its peers with respect to the length of the path generated, the number of turns made, and the number of collisions. A collision occurs when a path goes through an obstacle. The best chromosome will have a value of 1 in each of the three categories, indicating that it found the

TABLE I
APPROACHES EXAMINED

Citation	Name	Type	Chromosome Length	Fitness Function Measures
Sedighi et al., 2004 [35], 2009 [36]	Local path planner (LPP)	Monotone	Fixed	Turns, collisions and length of path
Sugihara & Smith, 1997 [28]	Adaptive motion planner (AMP)	Monotone	Fixed	Length of path and collisions.
Elshamli et al., 2004 [29]	Dynamic path planner (DPP)	Node Sequence	Variable	Distance between moves, smoothness of path and distance from obstacles.
Hu & Yang, 2004 [32]	Knowledge based path planner (KBPP)	Node Sequence	Variable	Distance between moves and depth of collision
Liu et al., 2004 [12]	Connected path planner (CPP)	Node Sequence	Variable	Number of disconnect paths (paths that hit obstacles) and length of valid paths.
Gombosi, 2001 [33]	Evolutionary navigator (EN)	Node Sequence	Variable	Distance between moves, length of chromosome, number of infeasible parts.
New	Dual-goal approach (DGA)	Node Sequence	Fixed	Distance to goal and length of path.

goal in the smallest number of steps, the fewest turns, and the least number of collisions. These values are input into the following equation:

$$f_{path}=f_{collision}\left|Lf_{Length}+Tf_{\#ofTurns}\right|(100/(L+T)). \quad (1)$$

where L and T are constants which have been assigned the values 1 and 2, respectively. When a path contains an obstacle, a penalty is imposed as follows:

$$f_{path}=0.1*(f_{path}/(\#ofcollisions)^2). \quad (2)$$

Although the best results were reported for search environments with two or fewer switching points, LPP's success rate was reported to be consistently better than the earlier methods on which it was based. Measuring the performance of each chromosome relative to its peers helps to ensure that the best chromosomes will survive into the next generation.

$$n*DistToGoal \quad (10)$$

Adaptive Motion Planner (AMP)

The adaptive motion planner [28] is another monotonic approach but, in this case, once a direction is selected, the entire route is either row-wise or column-wise. While the technique also allows the robot to move diagonally, our example allows only 90 degree angle turns.

The first value in this fixed length chromosome indicates whether travel is in a row- or column-wise direction. This indicator is followed by 16 2-value blocks where the first value indicates direction and the second the number of moves to make. This second value is only used if the move is on the same monotonic plane. For example, the partial chromosome in Table III represents the first 5 moves along a path, where the first value (0) represents a row-wise approach and the next pair of values (0,3) indicates the direction of movement and number of moves. If traveling row-wise and the first value is 0, the movement is horizontal to column 3 and then down one. The pair of values in the third position of the chromosome (1,x) indicate a one-position vertical move; in this case, because the direction of travel has changed, the second value is ignored. To facilitate obstacle avoidance, a slight modification has been made to the original in this study, allowing the robot to traverse both left and right along an axis. For example, when traveling to row 2, the column coordinate is read as 3, whereas in the original, the movement would have been three steps in the positive horizontal direction. See [28] for a fuller explanation.

For a path that avoids obstacles, the fitness calculation is as follows:

$$f_{path}=(1+w_{\max})^2-pathLength; \quad (3)$$

where the constant w_{max} is a weight applied to all solid objects. This constant was set to 4 in the original paper.

TABLE III
EXAMPLE OF PARTIAL CHROMOSOME FOR ADAPTIVE MOTION PLANNER (AMP)

Position	Value
R/C	0
2	0
	3
3	1
	x
4	1
	x
5	0
	7
6	0
	10
7	0
	11

The key advantage of this approach is that no specific operators are required to process the fixed length chromosomes. The initial study indicated that this technique is adaptable to changing environments.

B. Node Sequence Approaches

Node sequence approaches differ from the monotonic techniques in that the values in the chromosomes represent both the *x* and *y* coordinates, and each move can traverse multiple rows and/or columns. The only limit placed on the selected coordinates is that they must not contain obstacles, although a path between two coordinates may contain obstacles. The chromosome does not contain information on the direction of travel; therefore, each single step is first tested horizontally and, if unsuccessful (due to an obstacle), a vertical move is tried. Some of the approaches also use problem-specific operators such as repair and deletion operators to fix infeasible nodes [32, 33]. However, because our experiments were designed to focus solely on fitness functions, these special operators were not applied. This subject is revisited in our discussion.

Dynamic Path Planner (DPP)

DPP [31] uses a variable length chromosome that can be anywhere in length from 2 to n*2, where each pair of values represents one obstacle free location on the grid. The fitness function evaluates individual paths based on their length, smoothness, and clearance or closeness to obstacles. Smoothness attempts to find a path that has few turns and is defined as the curvature at a knot or intermediate node [14]. The equation for calculating smoothness can be found in [27]. The fitness function is as follows:

$$f_{path}=w_d dist(p)+w_s smooth(p)+w_c clear(p) \quad (4)$$

where w_d, w_s and w_c represent constant weight values. (Although the exact values were not given in the original paper, the weights applied in this study were 10, 2 and 50 respectively, to emphasize the goal of finding an adequate route with good clearance.) If a generated path is infeasible because of obstacles, the fitness value is the ratio of the number of feasible to infeasible segments.

This approach was reportedly effective in both static and dynamic environments for 10 runs but the size of the search space was not indicated. The DPP fitness function takes into consideration one of the key issues in path planning -- that is, an optimal path must be straight as well as short.

Knowledge Based Path Planner (KBPP)

KBPP [32] uses a slightly different approach to its chromosome formation. Instead of including separate x and y coordinates to indicate moves, a single value is first assigned to each position in the grid. The resulting chromosome can vary in length from 2 to n. The reported technique allowed diagonal moves between positions; however, in this study, movement is limited to 90 degree turns. The fitness function is as follows:

$$F_{cost} = \sum_{t=1}^{N} (d_i + \beta_i C) \quad (5)$$

where d is the Euclidean distance between two moves, C is a constant, and β is the coefficient denoting the depth of collision. Depth of collision is 0 if there are no collisions; otherwise, it is the number of moves from the current position to the closest feasible location.

Unlike the previous technique, this method uses only infeasibility and distance traveled as fitness measurements. The addition of the depth of collision would allow specific genetic operators to be written to amend a path. However, in some real-world applications the size of an obstacle is unknown or as in the case of long wall, measurements of depth may be meaningless.

Connected Path Planner (CPP)

CPP [11] uses a variable length chromosome where the intermediate nodes are represented by x and y coordinates and each node entry contains a pointer to the next logical node in the path. To establish these links, the *x* and *y* coordinates of each chromosome are sorted, row dominant, prior to testing on the grid. The purpose of this procedure is to make shorter, closer moves rather than jumping back and forth across the grid. The size of the variable length chromosome ranges from 2 to n*2. The fitness function, which is designed to minimize infeasible segments and produce short chromosomes, is defined as follows:

```
if (infeasibleSegments != 0) then          (6)
    fitness = 1/(infeasibles+1)
else
    fitness = 1+(1/chromosomeLength)
```

The variable, infeasibleSegments, refers to the number of infeasible path segments. When a feasible path is found, the fitness value reverts to the length of the chromosome.

This process places strong emphasis on short chromosomes, ignoring the actual length of the path. As a result, more longer moves through the grid may occur. This problem is offset by sorting the segments so that closer moves are made first. CPP was reportedly quite successful in simulations.

TABLE IV
PARTIAL CHROMOSOME FOR DUAL GOAL APPROACH

x	y
10	6
11	1

Evolutionary Navigator

EN [33] uses a variable length chromosome that is very similar to the DPP chromosome to represent the *x* and *y* coordinates. Each pair of coordinates is accompanied by a state variable to capture information about infeasible points and segments. The fitness function is defined as follows:

$$fitness = w_d * dist(p) + w_l * length(p) + w_i * inv(p) \quad (7)$$

where *dist* is the distance traveled, *length* is the length of the chromosome, and *inv* is the number of infeasible segments. Although the values of the three constants (w) were not specified in the original paper, they have been set to 10, 2, and 50 respectively for these experiments.

Like CPP, EN uses the chromosome length as part of the fitness function but combines it with the distance traveled. This may give it an advantage over CPP; but, in contrast to DBPP, it does not take into consideration the smoothness of the path, i.e., the number of turns. One of the cited advantages of this approach is that it offers effective tradeoffs between near-optimality of the path and planning efficiency.

Dual Goal Approach

The proposed Dual Goal Approach (DGA) uses a fixed length chromosome of size (n*2). In this approach, the path between pairs of *x,y* coordinates is examined and, if an obstacle interferes with the path, that segment is ignored and the next segment is tested.

For example, given the chromosome in Table IV, the first move would be from {1,1} to {10,6}; however, it is impossible to reach {10,6} without encountering an obstacle and this move is not made. The next move attempted is {1,1} to {11,1}, which is valid, and the move made. Therefore, while a chromosome may have many segments, the invalid segments are not used. Because testing invalid segments is time consuming, the fitness function penalizes a chromosome for invalid tests in an attempt to reduce their number. This approach is similar to the gene decomposition method proposed by [37].

After each valid move, a test is performed to determine whether there is a clear path from the current location to the goal. This avoids needless moves when the goal is within reach. Again, the chromosome is penalized for each goal check it must perform. A path is never allowed to travel through an obstacle and, if a path cannot find the goal without hitting an obstacle, it is given a low fitness value.

Fitness is assessed via a 2-step process that uses three fitness functions:

$$t * (1.0 / DistToGoal\ ) \quad (8)$$

If no previous chromosome has reached the goal, fitness is evaluated as follows to give higher fitness values to those that came closest to reaching the goal: where t is the number of successful moves made for this chromosome and *DistToGoal* is the Euclidean distance between the last valid position and the goal.

When a chromosome reaches the goal, a flag is raised and the fitness of all subsequent chromosomes is evaluated as follows:

a. If the chromosome has reached the goal, the fitness is:

$$1 + (1.0 / pLen(goalChecks + invSegs) * numTrns \quad (9)$$

where *pLen* is the Euclidean distance from the starting position through each intermediary node to the goal.

b. If the flag has been raised by a previous chromosome, but the current chromosome has not found the goal, the fitness is:

$$n * DistToGoal \quad (10)$$

where *n* is the size of the grid.

This method has two features that make it significantly different from the others. First, a move is not made that would travel through an infeasible location. Second, DGA rewards a chromosome for finding a clear path that comes close to the goal. Although similar to the work of [38], where only valid trajectories between start and goal are allowed, the approach presented here does not presume that a chromosome represents a valid path. Both of these features are unique among the fitness functions discussed here. Like the others, however, when the goal is found, the length of path is again used as a measurement of fitness.

IV. Experimental Studies

The sections that follow describe a series of experiments designed to evaluate the performance of the seven approaches relative to one another. The primary research question is:

Which path planning approach is best in terms of consistently finding a path from the starting position to the goal in the fewest number of moves (efficiency), while avoiding obstacles (reliability)?

In each of these experiments, the value encoding technique was used to encode the chromosomes, because the monotonic techniques require the genes' array positions in order to create a route from start to goal [30]**.** Value encoding is more natural than binary, and extensive testing indicates that it produces more consistent results across replications and consumes fewer resources in the process [38, 39]. To ensure comparability, the population size for all trials was set at 60, while the crossover and mutation rates were set at 0.80 and 0.03 respectively [41]. Selection was performed using the fitness-proportionate technique described by [27]. In the first experiment, the number of generations was restricted to 20; in the subsequent experiments, which used a much larger search space, the number of generations was set at 50.

In all experiments, reliability was evaluated based on the number of runs the GA completed; the measure of efficiency was the number of moves between the starting position and goal. "Completed runs" refers to the number of times the GA succeeded in producing an obstacle-free path, given the generational constraints established for each experiment. One-way analysis of variance (ANOVA), an extension of the t-test that compares multiple group means, was used to evaluate the significance of the observed differences between approaches in each experiment. In cases where the F-test was significant, pairwise comparisons were made using the Tukey test, a statistical procedure that determines if the difference between two treatments is due to the treatments themselves or if the difference is simply due to random chance. The Tukey test was chosen for these pairwise comparisons because the family-wise error rate is exactly equal to the assumed value of alpha [42]. In all cases, the significance level was set at 0.05.

TABLE V
SUMMARY RESULTS FOR THE 16X16 GRID.

	LPP	AMP	DPP	KBPP	CPP	EN	DGA
Mean Number of Moves	55.3	30.2	31.4	30.6	31.0	31.9	30.0
Median	56.0	30.0	30.0	30.0	30.0	30.0	30.0
Std Dev	10.6	0.9	4.3	3.1	3.6	6.3	0.0
Avg. Number of Turns	17.3	7.7	4.8	4.2	4.3	4.8	2.0
Completed Runs	51	100	100	100	100	100	100

TABLE VI
PAIRWISE COMPARISON OF MEAN NUMBER OF MOVES FOR 16X16 GRID.

	AMP	DPP	KBPP	CPP	EN	DGA
LPP	25.1	23.9	24.8	24.4	23.5	25.3
AMP		1.2	0.4	0.8	1.7	0.2
DPP			0.9	0.5	0.4	1.4
KBPP				0.9	1.3	0.9
CPP					0.9	1.0
EN						1.9

TABLE VII
RESULTS FOR 100X100 GRID 2

	Mean Moves	Comple-tions	Turns
LPP	2386	96	183.5
AMP	963	100	86.8
DPP	202	100	6.9
KBPP	203	98	7.3
CPP	276	100	9.9
EN	212	100	7.4
DGA	198	100	2.2

A. Demonstration 16x16 Grid

The first experiment used the small 16x16 grid shown in Fig. 1, which contains eight obstacles. A similar sized grid

was used in previous research [28]. The GA was run 100 times for each approach and each run was limited to 20 generations regardless of whether or not an obstacle-free path had been found. The results shown in Table V indicate that DGA consistently found a path in the fewest number of steps and turns, while LPP completed only 51 of its 100 runs. The ANOVA results ($F=223.41$, $p<0.0001$) led to rejection of the null hypothesis that all population means are equal.

The values in Table VI represent the absolute difference in the mean number of moves for each pair of approaches; shading indicates significant differences. To determine whether the observed differences are statistically significant, the Tukey test was performed on all possible pairs. In this case only LPP, the worst performer, was significantly different from the other six in terms of path length. The other monotonic approach, AMP, performed as well as the node sequence approaches; however, it produced a path with significantly more turns. The mean number of moves for DGA was approximately the same as the other six path lengths, but it consistently produced solutions with fewer turns.

B. Expanded 100x100 Grids

For the next series of experiments, the size of the grid was increased to 100x100 and four different obstacle configurations were tested (see Appendix A). Each approach was run for 50 generations, increased over the previous test because of the larger grid size. The start position was {1,1} and the goal {100,100}. All GA parameters remained the same, and the mean number of moves was again used as the measure of efficiency.

ANOVA was run on the results from each of these four trials, leading to rejection of the null hypothesis in each case. The monotone approaches produced consistently poorer results overall, as illustrated by the results for Grid 2, which are shown in Table VII.

LPP and AMP averaged 1674 moves when combined versus an average of 218 moves for the node sequence approaches. This is more than likely due to the column and row restrictions they impose, which produce initial routes with considerable back and forth movement. Given the generational restrictions, there was simply not enough time for

them to converge on a straighter path. Because efficiency is a primary concern in path planning and the monotone approaches performed no better on subsequent trials, their performance on the remaining three 100x100 grids is not reported here.

To isolate the significant differences in the node sequence approaches, pairwise Tukey tests were run. The results for three of the four 100x100 grids, shown in Table VIII, were remarkably consistent, with CPP performing significantly worse than the other node sequence techniques on grids 2, 3, and 4. CPP has a characteristic that makes it unique: it sorts the chromosomes prior to testing them to reduce back and forth motion and produce shorter, closer moves. While the resulting segments may be shorter, however, sometimes longer steps are needed to avoid obstacles, causing a reduction in its efficiency.

Not evident from the results shown in Table VIII are the completion rates. On grids 2, 3, and 4, all node sequence approaches completed each of the 100 runs except KBPP, which finished 98, 90, and 97 runs respectively, making it somewhat less reliable than the others. On grid 5, only EN and DGA completed all 100 runs, with DPP (97) and CPP (98) coming close. KBPP fell significantly behind on this grid, completing 78 of its runs.

C. Comparison of Node Sequence Approaches in a Mars Landscape and a Maze-like Environment

In the third set of experiments, the five node sequence approaches were compared using two real-world scenarios, the Mars landscape and a maze-like environment with multiple obstacles and narrow passages. Fig. 2 is a simplified the binary image from http://science.nationalgeographic.com/science/photos/mars/#/mars-blueberries_1066_600x450.jpg where only the larger obstacles must be avoided. Although the landscape image does not contain a large number of obstacles, it is the shape of the obstacles combined with the 600x600 grid size that poses challenges for path planners. One start and end position was tested and the arrows and line in the figure show one possible route.

The results shown in Table IX indicate that only two techniques, CPP and DGA, were consistently able to find

TABLE VIII
PAIRWISE COMPARISON OF MEAN NUMBER OF MOVES FOR 100X100 GRIDS.

Grid 2	**KBPP**	**CPP**	**EN**	**DGA**
DPP	0.2	72.9	9.0	4.9
KBPP		73.1	9.2	4.7
CPP			63.9	77.8
EN				13.8
Grid 3	**KBPP**	**CPP**	**EN**	**DGA**
DPP	9.5	119.5	2.4	10.4
KBPP		129.0	7.1	0.9
CPP			122.0	130.0
EN				8.0
Grid 4	**KBPP**	**CPP**	**EN**	**DGA**
DPP	9.9	43.5	11.5	15.6
KBPP		53.4	1. 7	5.7
CPP			55.0	59.0
EN				4.0
Grid 5	**KBPP**	**CPP**	**EN**	**DGA**
DPP	0.2	4.2	2.7	4.9
KBPP		3.9	2.5	5.2
CPP			1.4	9.1
EN				7.6
Shading indicates significant results				

a valid route within the 50 generation limit. DGA successfully completed all 100 runs, while CPP completed only 78. A T-test performed on the means shown in Table IX indicated that the two techniques were significantly different from one another (t=11.319, p<0.0001) with DGA requiring half as many moves as CPP.

Because robots are often used for search and rescue missions, finding routes around obstacles such as walls and through doorways is a common experimental task in robotics research. The maze-like environment in Fig. 3 is built on a 100x100 grid and indicated by numbers are the three different start and goal combinations tested. The first search, represented by the number '1', requires a path from the upper left at {1,1} to lower right {98,98}; the second is a path from {86,46} to {10,10} (upper left); and the third from {32,93} (middle right) to {68,65}. Of the three, path two should be the most difficult because there are many narrow passages to find along the route. While other paths could have been selected for these tests, these explore a range of path difficulty for this grid. Again each approach was run for 50 generations for three different searches.

DGA was most successful at this task, finding a route for grid 1 on all 100 runs; 68 times out of 100 for grid 2; and 99 times for grid 3. Additionally, the average length of these paths and the number of turns is quite conservative considering the size of the grid. However, the results in Table X show that the remaining approaches were unable to find the goal more than a few times and KBPP never found a goal. This rather poor performance is likely the result of the limited number of generations allowed.

Unlike the simple grids tested previously, these two problem types were significantly more challenging for the GAs. In the case of the Mars landscape, where CPP does fairly well relative to its peers, the sorting of the chromosomes prior to testing and/or the chromosome length-based fitness function may be better measures of fitness than distance traveled. However, in the maze-like problem, neither the distance traveled nor the chromosome length appears to give CPP an advantage when a path travels through infeasible locations. DGA which only uses feasible paths had much better results, finding a feasible path on most runs.

TABLE IX
RESULTS FOR MARS LANDSCAPE

	DPP	KBPP	CPP	EN	DGA
Mean Number of Moves	1788	1247	2265	-	1158
Median	1977	1247	2108	-	1215
Std Dev	589	-	938	-	245
Avg Number of Turns	13	10	16	-	4
Completed Runs	3	1	78	0	100

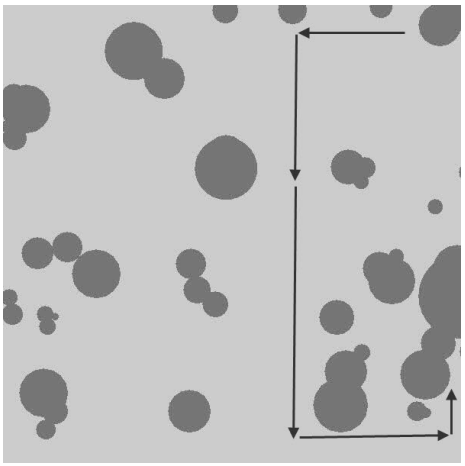


Figure 2. Simplified Binary Image of Mars Lunar Landscape http://tinyurl.com/d9hu55z

V. EXPERIMENT SUMMARY

The results of these experiments, which are summarized in Table XI, are not inconsistent with one another. Overall, the node sequence approaches outperformed the monotonic approaches. However, on more complex grids, DGA performed significantly better than the other node sequence methods. The goal-oriented feature of DGA, which measures an infeasible path from its point of infeasibility to the goal, appears to encourage feasible paths that are close to the goal over those that are far away. This feature enables the GA to quickly find a feasible path. On the other hand, techniques that measure the number of infeasible segments have a significant problem to overcome – while an infeasible path may contain just one obstacle, finding a way around the obstacle may be very difficult because a clear route could be quite far away. Another feature of DGA that differentiates it from the other node sequence approaches is the fact that it uses fixed length chromosomes, although only part of the chromosome may actually be used. The selective use of the chromosome segments ensures that only feasible moves are made, but a greater number of possible moves are produced. Testing for a direct path to the goal after each valid move, while potentially resource intensive, avoids inefficient moves away from the goal. Building valid move lists might reduce the potentially resource-intensive nature of this approach.

TABLE X
RESULTS FOR THE THREE SEARCHES WITHIN THE MAZE

		DPP	KBPP	CPP	EN	DGA
Route 1	Mean Moves	198	-	201	222	205
	Completions	2	0	2	1	100
Route 2	Mean Moves	-	-	-	-	247
	Completions	0	0	0	0	68
Route 3	Mean Moves	152	-	138	134	139
	Completions	1	0	2	2	99

Figure 3. Maze-like environment - numbers represent the three start and goal positions examined.

VI. Limitations

This study demonstrates the potential usefulness of GAs in motion planning; however, the limited context in which the seven approaches were tested restricts the generalizability of our findings in several ways. First, although a series of increasingly complex grids was used to evaluate the GAs, they do not represent all possible terrains and obstacle distributions. Further investigation using a variety of other layouts, including more real-world environments, would provide greater assurance as to the external validity of the results reported here. Second, the crossover and mutation settings in the GAs were held constant across methods and trials. Different settings could produce quite different outcomes. Third, these experiments used a point robot in a two-dimensional environment. Tests using an actual mobile robot are needed to confirm the validity of our experimental results.

Another concern is the internal validity of the experiments, i.e., the extent to which differences in the observed outcomes are in fact a result of the methods (chromosome structure and/or fitness functions) under test. In studies such as this one, internal validity could be affected by the selection of outcome measures or statistical analysis techniques [43]. Reliability was assessed based on the number of times a given approach found an obstacle free route; efficiency was assessed in terms of the number of moves needed to follow that route from start to goal. Similar investigations using other performance measures, (e.g., time or resource utilization) could lead to different conclusions as to the best fitness function for path planning. In addition, limitations were placed on the GAs to simulate the time and processing constraints that impinge on path planning in real-world applications. Additional experiments that vary the generational limit or the crossover and mutation rates would provide greater assurance of the internal validity of these findings.

Table XI
Summary of results

16x16 Grid	AMP, and the node sequence approaches performed equally well for this small grid.
4 - 100x100 Grids	The node sequence approaches performed significantly better than the monotonic approaches.
Mars Landscape	DGA significantly outperformed all other approaches
Maze	DGA significantly outperformed all other approaches

The statistical analyses were performed using standard analysis of variance (ANOVA), which assumes that the population means are normally distributed. In most of the trials reported above, the distribution is heavily skewed. Although skewness can make it difficult to detect true differences in the population means using one-way ANOVA, in the majority of the trials, the large number of data points and constant sample sizes would offset this shortcoming. Furthermore, ANOVA is considered robust with respect to both validity and efficiency, even when the underlying distribution of means is non-normal [42].

VII. Conclusions and Directions for Future Research

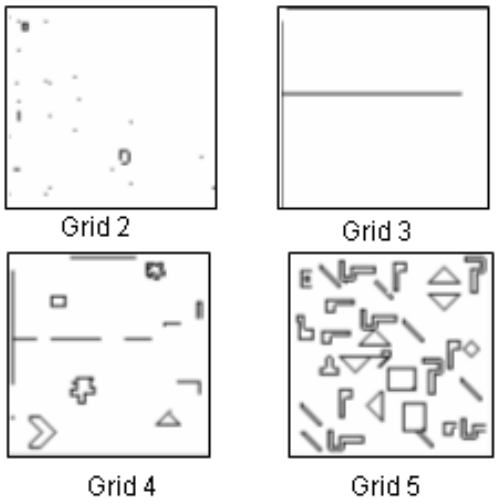
The results reported here were designed to assess a number of different GA approaches to path planning for mobile robots. Initially seven approaches were compared to one another. Then the best performers, the node sequence approaches, were tested in a more complex search space. The results suggest that GAs can be quite useful path planning tool for mobile robots, but the design of the fitness function will have an impact on the results, affecting both reliability and efficiency. On the smallest grid, most approaches performed reasonably well but, when the size of the grid was increased, the node sequence techniques outperformed the monotonic methods. In the most difficult task environments – the Mars landscape and a complex maze -- only the dual goal approach (DGA) had a consistently high success rate. A thorough evaluation of the components of the DGA fitness function would need to be performed in order to determine whether its success is a result of a single component, such as the fixed length chromosome, or the combination of components.

Currently the DGA approach is being used in a support capacity in our lab for a mapping mobile robot. It is used during the mapping process to generate an efficient backtracking route when the robot has encountered a dead-end or fully investigated a particular area. This technique could also be used to map routes between cooperating robots on search and rescue exercises, for example, should one robot need assistance. It is also possible for robots to work in parallel to generate the route.

This study provides a starting point for further empirical research focusing on path planning using GAs. Possible directions for future research include replicating this study in a real-world environment, amending the GA to work with multiple degrees of freedom, looking more

closely at the operators of the GA to determine whether it can be made more efficient, or comparison to non-GA-based path planning techniques.

APPENDIX A SAMPLE GRIDS



REFERENCES

[1] H. Zhang, Path planning methods of mobile robot based on soft computing technique, Advanced Materials Research,216 (2011) 677-680.

[2] G. Nagib and W. Gharieb, Path planning for a mobile robot using genetic algorithms, In Proceedings of the 2004 International Conference on Electrical , Electronic and Computer Engineering, Cairo, Egypt, (2004).

[3] J-C. Latombe, Robot Motion Planning, first ed., Klewer Academic, (1991).

[4] D. Whitley, A genetic algorithm tutorial. Statistics and Computing, 4 (1994), 65-85.

[5] M. Srinivas and, L.M. Patnaik, Genetic algorithms: A survey, IEEE Computer, 27 (1994), 17-26.

[6] J-W Chung, S-M. Oh, I-C Choi, A hybrid genetic algorithm for train sequencing in the Korean railway, OMEGA, 37 (2009) 555-565.

[7] A. Baresel, H. Sthamer, and M. Schmidt, Fitness function design to improve evolutionary structural testing, in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002). Morgan Kaufmann: New York, (2002), 1329-1336.

[8] D. Gallardo. and O. Colomina, A genetic algorithm for robot motion planning, in Proceedings of the Eleventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Castellos, Spain, (1998), 115 - 121.

[9] M. Gerke, Genetic path planning for mobile robots, in Proceedings of The 1999 American Control Conference, San Diego, CA, (1999), 2424-2429.

[10] C. Hocaoglu and A.C. Sanderson, Planning multiple paths with evolutionary speciation, *IEEE Transactions on Evolutionary Computation*, 5:3 (2001), 169 - 191.

[11] Y.K. Hwang and N. Ahuja, Gross motion planning—a survey, *ACM Computing Surveys*, vol. 24:3 (1992), 219-291.

[12] S. Liu, Y. Tian, and J. Liu, Multi mobile robot path planning based on genetic algorithm, In Proceedings of the 5th World Congress on Intelligent Control and Automation, Hangzhou, P.R. China, (2004).

[13] J. Tu. and S. Yang, Genetic algorithm based path planning for a mobile robot, In Proceedings of IEEE International Conference on Robotics and Automation, Taiwan, (2003).

[14] M. Wang and T. Wu, Cooperative co-evolution based distributed path planning of multiple robots. Journal of Zhejiang University *SCIENCE*, 6A:7 (2005), 697-706.

[15] J. Xiao, Z. Michalewicz, Z. Lixin, and K. Trojanowski, Adaptive evolutionary planner/navigator for mobile robots, *IEEE Transactions on Evolutionary Computation*, 1:1, (1997), 18-28.

[16] C. Zheng, L. Li, F. Xu, F, Sun, and M. Ding, Evolutionary route planner for unmanned air vehicles, *IEEE Transactions on Robotics and Automation*, 21:4, (2005), 609-620.

[17] O. Castillo, L. Trujillo, P. Melin, Multiple objective genetic algorithms for path-planning optimization in autonomous mobile robots, Soft Computing – A Fusion of Foundations, Methodologies and Applications, 11:2, (2007), 269 – 279.

[18] Y. Eun and Y. Bang, Cooperative task assignment/path planning of multiple unmanned aerial vehicles using genetic algorithms, Journal of Aircraft, 46:1 (2009), 338.

[19] L. Chen and C. Chiang, New approach to intelligent control systems with self-exploring process, IEEE Transactions on Systems, Man and Cybernetics, Part B, 33:1, (2003), 56- 66.

[20] N. Sadati. and J. Taheri, Genetic algorithm in robot path planning problem in crisp and fuzzified environments, In proceedings of IEEE International Conference on Industrial Technology (ICIT) , Bangkok Thailand, (2002).

[21] H.C. Chang,, Liu, J-S, High-quality path planning for autonomous mobile robots with n3-splines and parallel genetic algorithms' IEEE International Conference on Robotics and Biometrics, (2009), 1671-1677.

[22] C. Zeng, Quing Zhang, Z. Wei, GA-Based global path planning for mobile robot employing A* algorithm, Journal of Computers, Vol 7, No 2 (2012), 470-474.

[23] L.S. Sauter, M.S. Gottlieb, K.C. Jones, V.N. Dodson, and K.M. Rohrer, Job and health implications of VDT use: initial results of the Wisconsin-NIOSH study, *Communications of the ACM*, 26:4, (1983), 284-294.

[24] H. Seraji and A. Howard, Behavior-based robot navigation on challenging terrain: A fuzzy logic approach, *IEEE Transactions on Robotics and Automation*,18:3 (2002), 308-321.

[25] X. Luo and W. Wei, A new immune genetic algorithm and its application in redundant manipulator path planning, Journal of Robotic Systems, 21:3, (2004), 141-151.

[26] C. Wang, Y. C. Soh, H. Wang, and H. Wang , A hierarchical genetic algorithm for path planning in a static environment with obstacles, Canadian Conference on Electrical and Computer Engineering (CCECE 2002), Niagara Falls, NY, (2002), 1652-1657.

[27] H. Yu, A.S. Wu, K. Lin, and G. A. Schiavone, Adaptation of length in a nonstationary environment, In Proceedings of the Genetic and Evolutionary Computation Conference, (2003), 1541-1553.

[28] K. Sugihara and J. Smith, Genetic algorithms for adaptive motion planning of an autonomous mobile robot, In Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA'97), (1997), 138-143.

[29] P. McMinn, Search-based software test data generation: A survey, Software Testing, Verification and Reliability, 14(2004), 105-156, 2004.
[30] C.F. Lima, K. Sastry, D.E. Goldberg, and F.G. Lobo, Combining competent crossover and mutation operators: A probabilistic model building approach, In Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, H. Beyer, ed., Washington, DC, (2005), 735-742.
[31] A. Elshamli., H. Abdullah, and S. Areibi, Genetic algorithm for dynamic path planning, Canadian Conference on Electrical and Computer Engineering (CCECE 2004), Niagara Falls, Canada, (2004), 677-680.
[32] Y. Hu and S.X. Yang, A knowledge based genetic algorithm for path planning of a mobile robot, in Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, LA, (2004), 4350-4355.
[33] M. Gombosi, Evolution of path finding, In proceedings of the 23rd International Conference Information Technology Interfaces (ITI 2001), Pula, Croatia, (2001).
[34] T. Geisler and T. Manikas, Autonomous robot navigation system using a novel value encoded genetic algorithm, in Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems, Tulsa, OK, (2002).
[35] K. Sedighi., K. Ashenayi, and T. Manikas, Autonomous local path planning for a mobile robot using a genetic algorithm, in Proceedings of the 2004 IEEE Congress on Evolutionary Computation, Portland, OR, (2004).
[36] K. Sedighi, T. Manikas, K. Ashenayi, R. Wainwright, A genetic algorithm for autonomous navigation using variable-monotone paths, International Journal of Robotics and Automation, 24 (2009).
[37] A. Agogino, K. Tumer, and R. Miikkulainen, Efficient credit assignment through evaluation function decomposition, In Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, H. Beyer, Ed., Washington DC, USA,(2005), 1309-1316.
[38] V. Ayala-Ramirez, A. Perez-Garcia, F.J. Montecillo-Puente, and R.E. Sachez-Yanez, Path planning using genetic algorithms for mini-robotic tasks, In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics, The Hague, The Netherlands, (2004), 3746 – 3750.
[39] L. Davis, Ed., Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York:, (1991).
[40] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, third ed., Springer-Verlag, New York, (1991).
[41] A. Watkins and E.M. Hufnagel, Evolutionary test data generation: a comparison of fitness functions, Software - Practice and Experience, 36(2006), 95-106.
[42] S. Huck, *Reading Statistics and Research*, 4th Ed., Allyn & Bacon, Boston, (2003).
[43] D. Binkley, M. Harman, Analysis and visualization of predicate dependence on formal parameters and global variables, IEEE Transactions on Software Engineering, 30:10 (2004), 1-21.

Alison Watkins earned a Ph.D. in Computer and Information Systems from the University of Plymouth, United Kingdom in 1996 and an M.Sc. in Intelligent Systems from the same institution in 1992.

She is an Associate Professor of Information Systems at the University of South Florida St. Petersburg, Florida.