Intelligent Control and Planning of Autonomous Mobile Robots Using Fuzzy Logic and Genetic Algorithms

Julian Garibaldi, Azucena Barreras and Oscar Castillo

Dept. of Computer Science, Tijuana Institute of Technology, Mexico

Abstract. This paper describes the use of a Genetic Algorithm (GA) for the problem of Offline Point-to-Point Autonomous Mobile Robot Path Planning. The problem consist of generating "valid" paths or trajectories, for an Holonomic Robot to use to move from a starting position to a destination across a flat map of a terrain, represented by a two dimensional grid, with obstacles and dangerous ground that the Robot must evade. This means that the GA optimizes possible paths based on two criteria: length and difficulty.

1 Introduction

The problem of Mobile Robot Path Planning is one that has intrigued and has received much attention thru out the history of Robotics, since it's at the essence of what a mobile robot needs to be considered truly "autonomous". A Mobile Robot must be able to generate collision free paths to move from one location to another, and in order to truly show a level of intelligence these paths must be optimized under some criteria most important to the robot, the terrain and the problem given. GA's and evolutionary methods have extensively been used to solve the path planning problem, such as in (Xiao and Michalewicz, 2000) where a *CoEvolutionary* method is used to solve the path planning problem for two articulated robot arms, and in (Ajmal Deen Ali et. al., 2002) where they use a GA to solve the path planning problem in non-structured terrains for the particular application of planet exploration. In (Farritor and Dubowsky, 2002) an *Evolutionary Algorithm* is used for both off-line and on-line path planning using a linked list representation of paths, and (Sauter et. al., 2002) uses a *Particle swarm optimization* (PSO) method based on *Ant Colony Optimization* (ACO). However, the research work presented in this paper used as a basis for comparison and development the work done in (Sugihara, 1999). In this work, a grid representation of the terrain is used and different values

J. Garibaldi et al.: *Intelligent Control and Planning of Autonomous Mobile Robots Using Fuzzy Logic and Genetic Algorithms*, StudFuzz **208**, 255–265 (2007)
www.springerlink.com

are assigned to the cells in a grid, to represent different levels of difficulty that a robot would have to traverse a particular cell. Also they present a codification of all monotone paths for the solution of the path-planning problem.

2 Basic Theory

This section is intended to present some basic theory used to develop the GA's in this paper for use in the path planning problem, covering topics like basic Genetic Algorithm theory, Multi Objective optimization, Triggered Hypermutation and Autonomous Mobile Robot Point-to Point Path Planning.

2.1 Genetic Algorithms

A Genetic Algorithm is an evolutionary optimization method used to solve, in theory "any" possible optimization problem. A GA (Man et. al., 1999) is based on the idea that a solution to a particular optimization problem can be viewed as an *individual* and that these individual characteristics can be coded into a finite set of parameters. These parameters are the *genes* or the *genetic information* that makes up the *chromosome* that represents the real world structure of the individual, which in this case is a solution to a particular optimization problem. Because the GA is an evolutionary method, this means that a repetitive loop or a series of *generations* are used in order to evolve a *population S* of *p* individuals to find the *fittest* individual to solve a particular problem. The *fitness* of each *individual* is determined bye a given *fitness function* that evaluates the level of aptitude that a particular *individual* has to solve the given optimization problem. Each *generation* in the genetic search process produces a new set of individuals through *genetic operations* or *genetic operators: Crossover* and *Mutation,* operations that are governed by the *crossover rate* and the *mutation rate* μ respectively. These operators produce new *child chromosomes* with the intention of bettering the overall fitness of the population while maintaining a global search space. Individuals are selected for *genetic operations* using a *Selection method* that is intended to select the fittest individuals for the role of *parent chromosomes* in the *Crossover* and *Mutation* operations. Finally these newly generated child chromosomes are reinserted into the population using a *Replacement method.* This process is repeated a *k* number of *generations.*

2.2 Multiple Objective Genetic Algorithms

Real-world problem solving will commonly involve (Oliveira et. al., 2002) the optimization of two or more objectives at once, a consequence of this is that it's not always possible to reach an optimal solution with respect to all of the objectives evaluated individually. Historically a common method used to solve multi objective problems is by a linear combination of the objectives, in this way creating a single objective function to optimize (Sugihara, 1997) or by converting the objectives into restrictions imposed on the optimization problem. In regards to evolutionary computation, (Shaffer, 1985) proposed the first implementation for a multi objective evolutionary search. The proposed methods in (Fonseca and Fleming, 1993), (Srinivas, 1994) and (Goldberg, 1989), all center around the concept of *Pareto optimality* and the *Pareto optimal set*. Using these concepts of optimality of *individuals* evaluated under a multi objective problem, they each propose a *fitness* assignment to each individual in a current population during an evolutionary search based upon the concepts of *dominance* and *non-dominance* of *Pareto optimality*. Where the definition of *dominance* is stated as follows:

Definition 1: For an optimization (minimization) problem with n-objectives, solution u is said to be dominated by a solution v if:

$$\forall i = 1,2,\ldots,n. \qquad f_i(u) \geq f_i(v), \tag{1}$$

$$\exists j = 1,2,\ldots,n, \qquad \because\ f_i(u) > f_i(v) \tag{2}$$

2.3 Triggered Hypermutation

In order to improve on the convergence of a GA, there are several techniques available such as (Man et. al. 1999) expanding the memory of the GA in order to create a repertoire to respond to unexpected changes in the environment. Another technique used to improve the overall speed of convergence for a GA is the use of a Triggered Hypermutation Mechanism (Cobb, 1990), which consists of using *mutation* as a control parameter in order to improve performance in a dynamic environment. The GA is modified by adding a mechanism by which the value of μ is changed as a result of a dip in the fitness produced by the best solution in each generation in the genetic search. This way μ is increased to a high *Hypermutation* value each time the top fitness value of the population at generation k dips below some lower limit set beforehand.

2.4 Autonomous Mobile Robots

An Autonomous Mobile Robot as defined in (Xiao and Michalewicz, 2000) can be seen as a vehicle that needs the capability of generating collision free paths that take the robot from a starting position *s* to a final destination *d*, and needs to avoid obstacles present in the environment. The robot must be able to have enough relevant information of his current position relative to *s* and *d,* and of the state of the environment or terrain that surrounds it. One advantage about generating paths or trajectories for these kinds of robots, compared to the more traditional robot arms, is that in general there are far less restrictions in regards to the precision with which the paths must be generated. The basic systems that operate in an Autonomous Mobile robot are:

1. Vehicle Control.
2. Sensor and Vision.
3. Navigation
4. Path Planning

2.5 Point-to-Point Path Planning Problem

The path planning problem when analyzed with the point-to-point technique, (Choset et. al., 1999) comes down to finding a path from one point to another (start and destination). Obviously, one of the most important reasons to generate an appropriate path for a robot to follow, is to help it avoid possible danger or obstacles along the way, for this reason an appropriate representation of the terrain is needed generating a sufficiently complete map of the given surroundings that the robot will encounter along its route. The general path-planning problem, that all autonomous mobile robots will face, has been solved (to some level of satisfaction) with various techniques, besides the evolutionary or genetic search, such as, using the *Voroni Generalized Graph* (Choset et. al., 1999), or using a *Fuzzy Controller* (Kim et. al., 1999), yet another is by the use of *Artificial Potential Fields* (Planas et. al., 2002).

3 Proposed Method

The first step before we can continue and give the details of the GA implementation used to solve the path-planning problem, is to explicitly define the problem and what is it that we are expecting out of the subsequent genetic search. To this end, we propose what will be the *input/output* pair that we are expecting from our GA as follows:

Input: 1) An ***n*** x ***n*** grid, where the starting cell ***s*** for the robot is in one corner and the destination cell ***d*** is diagonally across from it.
2) Each cell with a corresponding *difficulty weight* ***wd*** assigned to it ranging from [0, 1].
Output: A path, defined as a sequence of adjacent cells joining ***s*** and ***d***, and that complies with the following restrictions and optimization criteria:
1) The path most not contain cells with ***wd*** = 0 (solid obstacles).
2) The path must stay inside of the grid boundaries.
3) Minimize the path length (number of cells).
4) Minimize the total difficulty for the path, that means, the combined values of ***wd*** for all the cells in a given path.

We must also establish a set of ground rules or assumptions that our GA will be operating under.
1) The ***n*** x ***n*** grid isn't limited to all cells in the grid having to represent a uniform or constant size in the terrain, each cell is merely a conceptual representation of spaces in a particular terrain.
2) Each cell in a terrain has a given *difficulty weight* ***wd*** between the values of [0,1], that represents the level of difficulty that a robot would have to pass through it, where the lower bounds 0 represents a completely free space and the higher bounds 1 represents a solid impenetrable obstacle.
3) The terrain is considered to be static in nature.
4) It is assumed that there is a sufficiently complete knowledge in regards to the state of the terrain in which the robot will operate.
5) The paths produced by the GA are all monotone paths.

4 Architecture of the Genetic Algorithm

We now turn to the actual implementation of our GA, used to solve the path-planning problem for one and two optimization objectives. So we describe each of the parts of our GA and give a brief description of each, clearly stating any differences between the one and two optimization objectives implementations.

4.1 Individual Representation

Basically, the chromosome structure was taken from the work done in (Sugihara, 1999) where a binary string representation of monotone paths is used. The binary string chromosome is made up of ***n***-1 (where ***n*** is the number of columns and rows in the grid representing the map of a given terrain) pairs of *direction/distance* of length *3 + log[2]**n**,* and an extra bit *a* which determines if the path is *x-monotone* (*a=0*) or *y-monotone* (*a=1*).

And each pair of *direction/distance* codes the direction in which a robot moves inside the grid and the number of cells it moves thru in that direction. The coding used greatly facilitates its use in a GA, because of its constant length no special or revamped genetic operators are needed, a problem that would be very cumbersome to solve if using a linked list chromosome representation of the path as done in (Xiao and Michalewicz, 2000).

4.2 Initial Population

The population *S* used in the genetic search is initialized with *p* total individuals. Of the *p* individuals in *S, p-2* of them are generated randomly while the remaining two represent straight line paths from *s* to *d,* one of this paths is *x-monotone* and the other is *y-monotone.*

So we can clearly define the population *S* as being made up by:

$$S = \{\, x_0, x_1, x_2 \ldots\ldots\ldots\ldots x_{p-2}, a, b \,\} \tag{3}$$

Where x_i are randomly generated individuals, and by *a* and *b* that are *x-monotone* and *y-monotone* paths respectively that take a straight-line route from *s* to *d.*

4.3 Path Repair Mechanism

Each path inside of the population *S* is said to be either *valid* or *non-valid.* Where criteria for *non-validity* are:

- Path contains a cell with a solid obstacle (***wd*** = 1).
- Path contains cells out of bounds.
- The paths final cell isn’t *d.*

Using this set of three rules to determine the state of validity of a given path for a particular genetic search, we can define a *subpopulation S’,* which is made up by entirely *non-valid* paths in *S.*

The Path Repair Mechanism used with the GA is a *Lamarckian* process designed to take *non-valid x’,* where *x’*   *S’,* and determine if they can be salvaged and return to a *valid* state, so as to be productive in the genetic search, because just because a particular path is determined to be *non-valid* this does not preclude it from having possible information coded in its chromosome that could prove to be crucial and effective in the genetic search process, this is way *non-valid* paths are given low fitness values with the penalty scheme used in the fitness evaluation, only after it has been determined that its *non-valid* state cant be reversed.

4.4 Fitness Evaluation

As was mentioned earlier, we introduce here both single and two objective optimization of the path planning problem, taking into account the length a given path and the difficulty of the same as the two criteria for optimization for paths in the population hence, the way in which each implementation of the GA assigns fitness values differs for obvious reasons.

4.4.1 Single Objective

Considering our Conventional GA, we can say that for paths inside *S* we optimize for only one objective, which is the path length, therefore we define fitness $f_1(x)$ as given by:

$$f_1(x) = (n^2) - (c) \tag{4}$$

Where *c* is the number of cells in a given path *x*.

4.4.2 Multiple Objective

Besides the fitness $f_1(x)$ used in Section 4.4.1 given for path length, a second fitness assignment $f_2(x)$ is given for path difficulty is given, and is calculated by,

$$f_2(x) = (n^2) - \Sigma wd_i \tag{5}$$

Where the second term in (5) is the sum of ***wd*** for each cell in a given path *x*. With this we are forced to use Pareto optimality for a rank-based system for individuals in population *S*. So for a path *x* where *x*   *S* its final fitness values is given by their *rank* value inside of *S* determined by,

$$\text{rank(x)} = \text{p - t} \tag{6}$$

Where *p* is the size of population *S* and *t* is the number of individuals that dominate *x* in *S*.

5 Simulation Results

We use the benchmark test presented in Figure 1, which was used in (Sugihara, 1997) due to its capability of projecting an accurate general performance score for the GA, and the performance measure of *probability optimality* $L_{opt}(k)$, which is a representation of the probability that a GA has of finding an optimal solution to a given problem. In this case, is the probability of finding a solution on the Pareto optimal front. Using

$L_{opt}(k)$ as the performance measure we present a set of optimal operating parameters for our MOGA using both a Generational and Elitist replacement scheme, Figures 2 to 3 show the simulation results that support this values. We also compare the two methods along with the GA proposed in (Sugihara, 1999) and the comparison is made under a normalized value for *kp=30,000* keeping the overall computational cost equal for each GA.

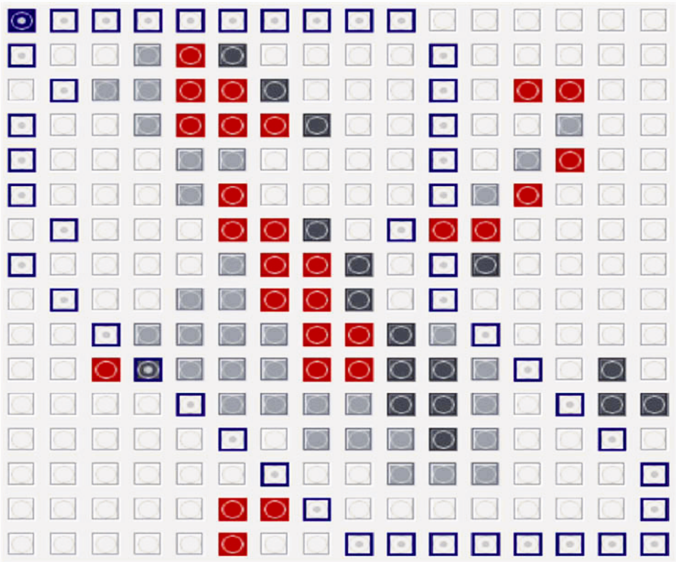

Fig. 1. Benchmark Test, with two paths on the Pareto Optimal Front.

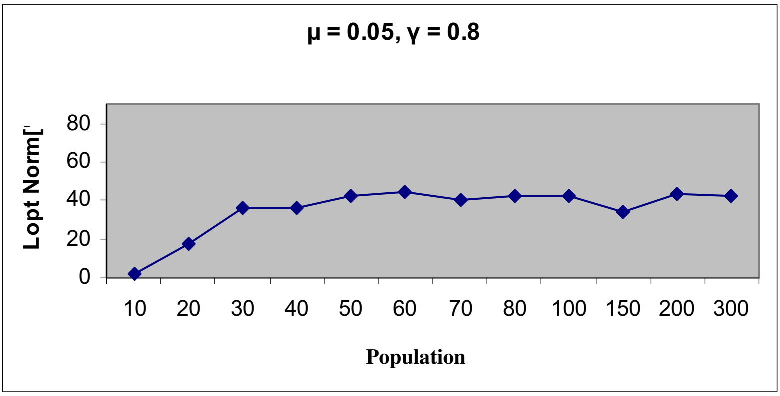

Fig. 2. Normalized $L_{opt}(k)$ and population size with Generational Replacement

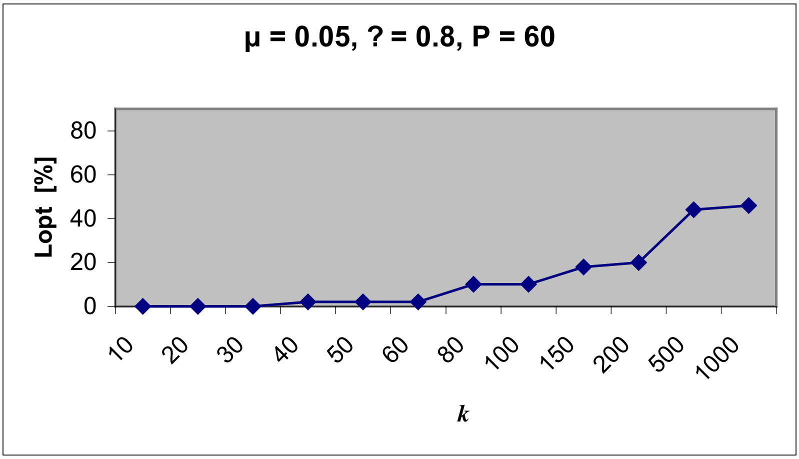

Fig. 3. $L_{opt}(k)$ and number of generations with Generational Replacement.

6 Conclusions

This paper presented a GA designed to solve the Mobile Robot Path Planning Problem. We showed with simulation results that both a Conventional GA and a MOGA, based on Pareto optimality, equipped with a basic repair mechanism for *non-valid* paths, can solve the point-to-point path planning problem when applied to grid representations of binary and continuous simulation of terrains respectively. From the simulation results gathered from experimental testing the Conventional GA with a Generational Replacement scheme and Triggered Hypermutation (which is commonly referred to as a conversion mechanism for dynamic environments) gave consistent performance to varying degrees of granularity in the representation of terrains with out a significant increase in population size or number of generations needed in order to complete the search in a satisfactory manner, while the MOGA based on Pareto Optimality combined with a Elitist replacement scheme clearly improves upon previous (Sugihara, 1999) work done with multiple objective path planning problem based on linear combination, with the added advantage of providing more than one equally usable solution.

References

Xiao, J. and Michalewicz, Z. (2000), An Evolutionary Computation Approach to Robot Planning and Navigation, in Hirota, K. and Fukuda, T. (eds.), *Soft Computing in Mechatronics*, Springer-Verlag, Heidelberg, Germany, 117 – 128.

Ajmal Deen Ali, M. S., Babu, N. and Varghese, K. (2002), Offline Path Planning of cooperative manipulators using Co-Evolutionary Genetic Algorithm, *Proceedings of the International Symposium on Automation and Robotics in Construction*, 19th (ISARC), 415-124.

Farritor, S. and Dubowsky, S. (2002), A Genetic Planning Method and its Application to Planetary Exploration, *ASME Journal of Dynamic Systems, Measurement and Control*, **124**(4), 698-701.

Sauter, J. A., Matthews, R., Parunak, H. V. D. and Brueckner, S. (2002), Evolving Adaptive Pheromone Path Planning Mechanisms, *First International Conference on Autonomous Agents and Multi-Agent Systems*, Bologna, Italy, 434-440.

Sugihara, K. (1999), Genetic Algorithms for Adaptive Planning of Path and Trajectory of a Mobile Robot in 2D Terrains, *IEICE Trans. Inf. & Syst.*, Vol. E82-D, 309 – 313.

Sugihara, K. (1997), *A Case Study on Tuning of Genetic Algorithms by Using Performance Evaluation Based on Experimental Design*, Tech. Rep. ICS-TR-97-01, Dept. of Information and Computer Sciences, Univ. of Hawaii at Manoa.

Sugihara, K. (1997), Measures for Performance Evaluation of Genetic Algorithms, *Proc. 3rd. joint Conference on Information Sciences*, Research Triangle Park, NC, vol. I, 172-175.

Cobb, H. G. (1990), An Investigation into the Use of Hypermutation as an Adaptive Operator in Genetic Algorithms Having Continuous, Time-Dependent Nonstationary Environments, Technical Report AIC-90-001, Naval Research Laboratory, Washington, D. C.

Fonseca, C. M. and Fleming, C. J. (1993), Genetic algorithms for multiobjective optimization: formulation, discussion and generalization, *5th Int. Conf. Genetic Algorithms*, 416-423.

Srinivas, M. and Deb, K. (1994), Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms, *Evolutionary Computation*, **2**(3), 221-248.

Man, K. F., Tang, K. S. and Kwong, S. (1999), *Genetic Algorithms*, Ed. Springer, 1st Edition, London, UK.

Oliveira, G. M. B., Bortot, J. C. and De Oliveira, P. P. B. (2002), Multiobjective Evolutionary Search for One-Dimensional Cellular Automata in the Density Classification Task, in Proceedings of *Artificial Life VIII*, MIT Press, 202-206.

Schaffer, J. D. (1985), Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms, 93-100.

Goldberg, D. E. (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA.

Spandl, H. (1992), *Lernverfahren zur Unterstützung der Routenplanung fur eine Mobilen Roboter*, Diss. Universität Karlsruhe, auch VDI-Forsch-Heft Reihe 10 Nr. 212, Düsseldorf, Germany, VDI-Verlag.

Choset, H., La Civita, M. L. and Park, J. C. (1999), Path Planning between Two Points for a Robot Experiencing Localization Error in Known and Unknown Environments, *Proceedings of the Conference on Field and Service Robotics (FSR'99)*, Pittsburgh, PA.

Kim, B. N., Kwon, O. S., Kim, K. J., Lee, E. H. and Hong, S. H. (1999), "A Study on Path Planning for Mobile Robot Based on Fuzzy Logic Controller", *Proceedings of IEEE TENCON'99*, 1-6.

Planas, R. M., Fuertes, J. M. and Martinez, A. B. (2002), Qualitative Approach for Mobile Robot Path Planning based on Potential Field Methods, *Sixteenth International Workshop on Qualitative Reasoning (QR'02)*, 1 -7.