A GENETIC ALGORITHM FOR PATH PLANNING

Ian J Griffiths, Qasim H Mehdi, Tingkai Wang and Norman E Gough

*School of Computing and Information Technology, University Of Wolverhampton,
Wolverhampton, UK, WV1 1SB
ex1131@wlv.ac.uk*

Abstract: This paper details work on the development of an path planning system for an automated guided vehicle. An evolutionary approach, using a genetic algorithm, is taken which uses an interpretation scheme to ensure that all generated solutions are legal. Results are presented which show that this approach has the ability to plan a route between two points that is optimal in terms of both distance travelled and time. The algorithm is designed so that it could be embedded within a microcontroller and provide some intelligent on board control.

Keywords: Path planning, Autonomous guided vehicles, Genetic algorithms, Microcomputers

1. INTRODUCTION

Current research on autonomous guided vehicles (AGVs) has attracted considerable attention in the areas of mechanical engineering, electronic engineering, computer science, information science, biology and psychology due to AGVs' great potential in applications. Possible applications include automated manufacturing, surveillance, unmanned exploration of the oceans and space amongst others.

Evolution is an exercise in optimisation - a selection process to produce the best possible design from a large number of possibilities. Now man has begun to apply nature's method to optimisation problems using genetic algorithms. The resulting techniques are extremely powerful, with enormous potential in a very wide range of applications.

Path planning is a difficult task for AGVs. It normally involves finding the optimal route between a start point and an end point. Optimisation of the route could involve several factors: minimum time, minimum distance or partially required routes, i.e. the necessity for part of the path to take the AGV to selected way points. The path planning could also involve other complications such as pedestrian or AGV traffic and the placing and removal of obstacles.

The University of Wolverhampton's AGV Research Group is interested in the design of AGVs that can be used in a variety of environments. Previous work on vision systems (Abu-Alola *et al.*, 1994; Gough *et al.*, 1994; Griffiths *et al.* 1997), navigation and path planning (Wang *et al.*, 1996a; Wang *et al.*, 1996b; Wang *et al.*, 1996c) and genetic algorithms has drawn attention to the necessity and complexity of a robust path planning method. This work is the initial formulation of a path planning method utilising genetic algorithms.

The organisation of the paper is as follows. Section 2 is an introduction to path planning. Section 3 is an introduction to path planning and also details the basic concepts of a simple genetic algorithm including populations, operators and fitness. An outline of the problem is presented in Section 4. It details the genetic algorithm used in this work and gives further details upon the genetic algorithm specific to this work. Section 5 presents results generated by the use of the genetic algorithm in path planning in the presence and absence of obstacles. Conclusions and further work are given in Section 6.

2. PATH PLANNING

Path planning is a difficult and necessary problem in AGV technology. The description of the problem and solutions vary, but there are typical path planning approaches and they are outlined here. The *virtual force field method* (Borenstein and Koren, 1989) uses a two-dimensional Cartesian histogram grid as a world model which is updated continuously with range data sampled by on-board range sensors. The virtual repulsion forces generated from obstacles and the virtual attractive force generated from the target are obtained according to the histogram grid. The AGV is pushed away from the obstacles by the repulsion forces, and pulled toward the target by the attractive force. A disadvantage with this method is that at some points a zero net force will be present leaving the AGV unable to move. Another method, *the edge detection method* (Borenstein and Koren, 1988), utilises the determination of two vertical visible edges on obstacles from sensor data to steer. The horizontal line connecting the two vertical edges are considered to be the boundaries of the obstacle and the AGV is steered around one side of the visible edges. A drawback of this method, however, is the sensitivity of the method to sensor accuracy and frequent misreading of ultrasonic sensors will make the performance unstable. *Wall following methods* (Bauzil *et al.*, 1981) are used by following the contour of an obstacle an AGV encounters until the AGV has negotiated the obstacle. The *fuzzy reasoning method* (Kinamura *et al.*, 1993; Lee and Wang, 1994; Wang and Mended, 1992) is based upon fuzzy mathematics and the reasoning is based upon fuzzy logic. The knowledge base of this method comes from human experiences in path planning that are represented as a degree of uncertainty and are realised via expert system methodology. It is an efficient method, but is prone to certain weaknesses. Firstly the knowledge and intelligence level greatly depend on the designer's experience, intelligence and ability and secondly updating knowledge is rather difficult i.e. the system can not be self learning and so it will make the same mistakes over and over again. Based upon the imitation of neural elements in the functioning of the human brain, *the neural network method* (Lumbers and Pander, 1994) possesses the ability of non-linear mapping and self-learning. The optimal or nearly optimal paths are sought via the learning process. This method does have drawbacks. Firstly there is currently a shortage of systematic theory to guide the design of neural network architecture. The current method for such design is dependent upon experience. Secondly the learning algorithm cannot guarantee a global optimal result. Finally the learning algorithm often takes a long time and affects the on-line operation of the system. The *hierarchical strategy* (Suh and Shin, 1988) employs the search strategies in AI, arranges all the possible routes into a layered tree-like structure so that the optimisation or near optimisation of paths is achieved. It may only be used for target searching. The feature of this method is similar to an expert system approach. The efficiency of the search strategy depends on the search technique and the structure of the tree. A final method to consider is the genetics-based approach . The approach uses human cell inheritance theory, via control of the inheritance cells by discarding bad cells while preserving good cells. This method is used to choose the optimal plans from multiple plans.

3. GENETIC ALGORITHMS

Genetic algorithms are computer programs that offer a robust solution to optimisation problems that have hitherto proved to be difficult or intractable (Schneider, 1989) and they are proving invaluable in industrial settings (Hughes, 1990). In control systems they have been successfully used to select optimal actuators (Rao *et al.*, 1991), to improve the performance of fuzzy logic controllers and to provide optimal discrete time control. They have also been used to optimise electronic design (Michalewicz *et al.*, 1990; Hulin, 1989), as a means for machine learning (Grefensette, 1988; Booker, 1988; Robertson and Riolo, 1989; DeJong, 1988), and within computer aided manufacturing (Goldberg and Holland, 1988).

Genetic algorithms have also been successfully applied to robotics. Examples include solving inverse kinematics for redundant robots (Kawakami *et al.*, 1991), obstacle avoidance with redundant robot manipulators (Parker *et al.*, 1989), the design of robot manipulators (Khoogar and Parker, 1991), and locomotion utilising neural networks .

Path planning is an important task for optimal motion of a robot in a structured or unstructured environment. Chung and Lee (1991) show how to plan the shortest collision-free path in 3-D, when a robot is navigated to pick up some tools or to repair some parts from various locations.

3.1. Basic Concepts

A genetic algorithm is a powerful search and optimisation technique that is modelled upon the processes of natural selection that appear in the natural world. In essence utilising a genetic algorithm involves producing a *population (pool)* of randomly generated *individuals (strings)*, where each individual refers to a specific solution of the problem, and manipulating this pool via a set of operators in order to evolve, after numerous generations, a solution that meets a set of criteria. The process is driven by means of the concept of *fitness*. Each individual has a fitness, a value determined by an *objective function*, and individuals contribute to the next generation according to their fitness via a *roulette wheel selection*. More fit individuals have a greater chance of providing information to the next generation than other less fit individuals. The basic concepts behind simple genetic algorithms (Goldberg, 1989) are shown in Fig. 1.

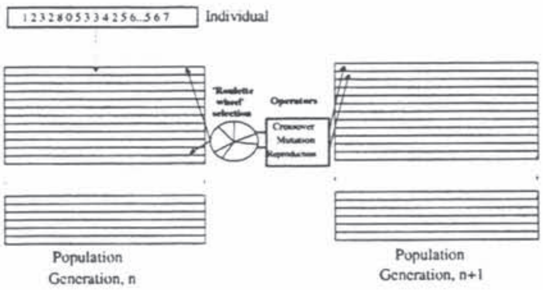

Fig. 1 A Simple genetic algorithm

3.2. Genetic Operators

The operators most commonly used are reproduction, crossover and mutation. Each perform a task of operating upon an individual or individuals. Reproduction takes a pair of individuals and carries them unaltered to the next generation. Crossover exchanges information between a pair of selected individuals as shown in Fig. 2. It works upon the basis that during evolution no individual will contain the ideal solution, but many individuals will contain some part of the ideal solution. Thus crossover will lead to individuals receiving portions of the ideal solution from their parents. The mutation operator, shown in Fig. 3, plays an important role in genetic algorithms as it ensures that highly fit individuals will not dominate and result in an incomplete search of the solution-space and from there the search will end in stagnation. This combination of operators is simple and effective.

Crossover

Parent 1 1 2 3 6 2 3 1 3 → 1 2 3 6 2 0 0 8 Child 1
Parent 2 4 5 6 4 2 0 0 8 → 4 5 6 4 2 3 1 3 Child 2

Fig. 2 The crossover operator

Mutation

1 3 4 2 $\boxed{3}$ 4 5 5 $\Rightarrow$ 1 3 4 2 $\boxed{0}$ 4 5 5

Fig. 3 The mutation operator

4. PROBLEM OUTLINE

It is necessary to navigate an AGV through an environment that contains numerous obstacles that the AGV should avoid. The AGV should negotiate a route from its start point to the necessary end point in the minimum possible time. It is also necessary to accommodate within the solution the capability to optimise these paths for multiple AGVs with multiple way, start and end points.

4.1 Coding of the solutions

Each individual of the population is a fixed length string that is made up of values of a nine symbol alphabet. The necessary coding and decoding to translate string values to movements is shown in Table 1.

Table 1 Decoding the solutions in the absence of obstacles

Value	Interpretation
0	N
1	E
2	S
3	W
4	NE
5	SE
6	SW
7	NW
8	WAIT

Thus the string can be interpreted as a series of moves, each move taking the AGV a unit distance in the direction indicated by the value. It is necessary to involve the WAIT value for two reasons. Firstly as an individual is a string of fixed length it details a fixed number series of moves. Using the WAIT command allows the genetic algorithm to be used without the concern of ensuring the string length exactly matches the necessary minimum number of moves. This ensures that if the AGV reaches its goal in less than the prescribed number of moves, the AGV is allowed to wait there , rather than be forced to move by remaining moves. Secondly it is necessary to consider that other AGVs may be present. This WAIT command will aid in the control of traffic flow.

A modification to this coding system is necessary where the AGV may possibly encounter obstacles. In the event that the AGV encounters obstacles an interpretation could lead to the AGV taking an illegal move by entering an area that contains an obstacle. Three possible methods of decoding were examined and are shown in Fig. 4.

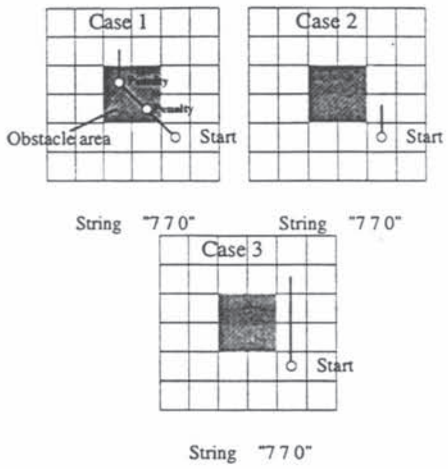

Fig. 4 Possible decoding techniques

Case 1: Penalty. The AGV has a penalty adjustment within the objective function that is applied when the decoding of the string produces a route whereby the AGV moves into an area of space that is an obstacle i.e. an illegal move. The difficulty in using this method is the application of the penalty value. This value will be dependent upon the environment as the value of a penalty over a large route will necessarily be larger than the value over a short route. Near perfect solutions in the latter case will be destroyed due to few penalties and these will not contribute to further generations. This leads to convergence of the algorithm to less than optimal solutions.

Case 2: No move. If an illegal move is generated during the decoding of a string the illegal move is interpreted as a WAIT command, but without the bonus that is associated with the WAIT command as this is concerned with optimising the time taken to reach the goal and not with the legality of moves. The problem with this method is that in an environment populated densely with obstacles the algorithm will take longer to converge to a solution and this solution may not be optimal.

Case 3: Alter direction. This was found to be the best decoding technique. Any illegal move is reinterpreted by taking the next legal move in a clockwise sense. The actual value of the move is not altered, but just it's interpretation. This means that the series of moves always has a legal sense and no interpretation of an individual can result in the AGV entering an illegal space. This is the method upon which the results presented in Section 5 have used.

4.2. The Objective Function

The objective function for the genetic algorithm is of the form :

$$J = 1 - \frac{d}{100} + \frac{w}{100} \quad (1)$$

where d is the distance between the final endpoint after performing the planned path and the required end point, w is the number of WAIT commands present in the planned path, and J is the fitness of the particular individual.

This objective function matches the Case 3 decoding. The exact form of the fitness function is the result of extensive testing necessary as there is no absolute equation to describe the fitness of solutions .

5. RESULTS

Results presented in this section are concerned with the use of the genetic algorithm to plan routes through two environments. The first environment shows the evolution of a path where the AGV will not encounter any obstacles. The second environment shows the AGV quickly negotiating an environment that contains obstacles. The computer code for these simulations was written in the C computer language.

5.1. Without obstacles

The algorithm here utilises a population of 120 individuals each having a string length of 10. The operators and there respective probabilities are crossover 0.6, mutation 0.1 and reproduction 0.3. The start point co-ordinate is (0,0), the end point co-ordinate is (10,10) and the environment contains no obstacles. The results of a run of the genetic algorithm are given in Table 2 and plots of the best of generation paths are shown in Fig. 5.

Table 2 Results of AGV GA in obstacle-free environment

Gen	Fittest string	End	Fit.	Ave.
0	2801802122	6,4	1.489	0.744
5	1201712114	6,6	1.558	0.817
10	1201711112	7,8	1.596	0.866
15	1101711111	7,10	1.361	0.901
20	1101211111	9,9	1.600	0.929
61	1111211111	10,9	1.201	0.970
118	1111111111	10,10	1.493	0.975

In Fig.5 the heavy circles upon the paths indicate the presence of a WAIT command, the number indicating the number of consecutive WAITS. These results clearly indicate the ability of the genetic algorithm to plan a path in an obstacle-free environment in minimum time. The presence of superfluous commands such as WAIT quickly disappear (in less than 5 generations) and the dominance of the most efficient command is apparent from the fifth generation.

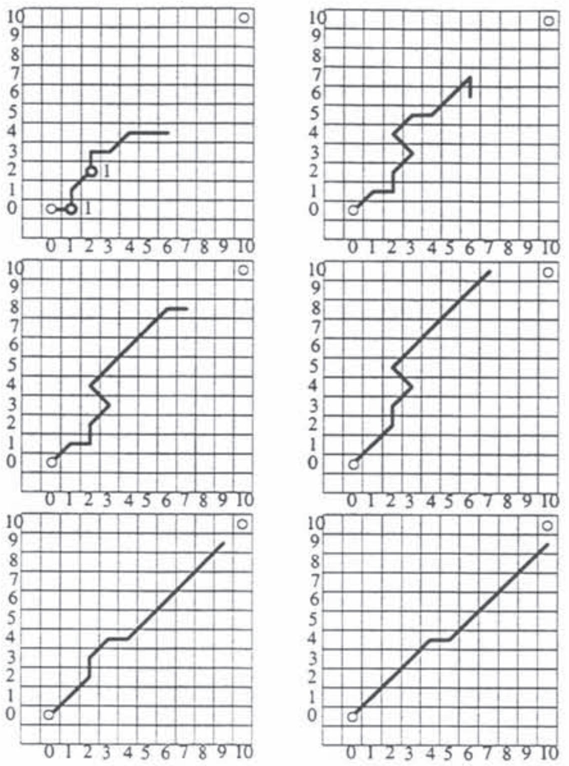

Fig.5 Best of generation paths shown at selected generations.

5.2. *With obstacles*

The algorithm here utilises a population of 120 individuals each having a string length of 10. The operators and their respective probabilities are crossover 0.6, mutation 0.1 and reproduction 0.3. The start point co-ordinate is (4,3), the end point co-ordinate is (4,8) and the environment contains obstacles. The results of a run of the genetic algorithm are given in Table 3 and the best of generation paths are shown in Fig. 6.

The paths shown in Fig. 6 are interpreted as in the previous section. The actual end point that the AGV reaches is shown as a shaded circle for clarity. It is apparent by using the coding technique of Case 3 where the legality of all paths is ensured the algorithm reaches it's goal quickly. Even at generation 0 where the best of generation path is made up of a purely randomly generated individual, the method allows the population to consist of individuals that are similar to the required solution. Through the evolutionary scheme the paths improve as shown by the increasing number of WAIT(8) commands that are shown in evidence (the required path is smaller than the fixed individuals' string length) and by reducing the number of unnecessary moves i.e. retracing parts of the route already covered (as shown in generation 0, generation 2 and generation 8). By the final illustrated generation the algorithm has converged to the optimal solution where obstacles are negotiated in the minimum possible time. These results show the robustness, ease and speed with which the genetic algorithm using the specialised case of path interpretation that ensure legality of paths can provide optimal routes to a simple obstacle populated environment.

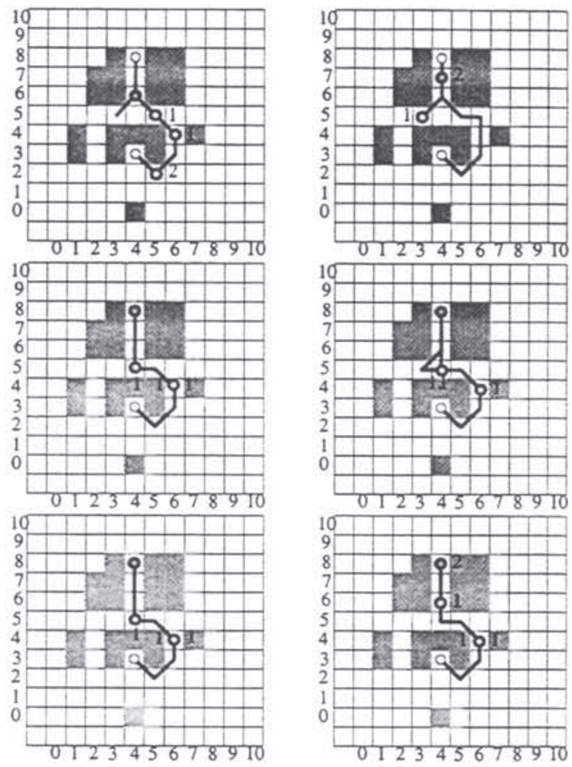

Fig. 6 Best of generation paths shown at selected generations.

Table 3 Results of AGV GA in environment with obstacles

Gen	Fittest string	End	Fit	Ave.
0	388158770318757	4,6	1.735	0.877
1	388158770318757	4,6	1.606	0.901
2	270077580702088	4,7	1.637	0.909
4	217863856715787	4,7	1.651	0.930
6	217863856718766	4,8	1.782	0.941
8	217863874850058	4,8	1.612	0.959
10	217863856718766	4,8	1.851	0.963
20	107863856780088	4,8	1.507	1.008
65	217863886780088	4,8	1.137	1.056
100	207863886780088	4,8	1.079	1.059
200	217868886780088	4,8	1.232	1.056
238	217868888780688	4,8	1.261	1.056

6. CONCLUSIONS

This work has been concerned with the development of a path planning system for AGVs that would allow the AGV to negotiate an environment that is populated with obstacles in an efficient and optimal manner. The simulations carried out and presented in this paper show the genetic algorithm that has been developed can carry out this task by using an interpretation of strings which ensures that each path evolved is legal i.e. does not enter an area that is defined as an obstacle.

With this work completed the next objective is to utilise previous work carried out upon trajectory planning, fuzzy and non-fuzzy embedded systems, and fuzzy-tuned stochastic transition matrices to build a working prototype of an AGV that can negotiate an environment using global and on-board sensor systems.

REFERENCES

Abu-Alola, A. et al (1994),Application of a genetic algorithm to an actuating system for robotic vision, *Proc. IEEE Int. Conf. on Control*, University of Warwick.

Bauzil,G. et al (1981), A navigation subsystem using ultrasonic sensors for the robot Hilare, *Proc. 1st Int. Conf. Robot vision and sensory controls*, Standford-upon-Avon, pp 47-58

Booker, L.B. (1988) Classifier systems that learn internal world models, *Mach. Learn.*,**3**, pp161-92.

Borenstein, J. and Y .Koren (1988) Obstacle avoidance with ultrasonic sensors, *IEEE J. Rob. Autom.*, RA-4(**2**), pp213-218.

Borenstein, J. and Y. Koren (1989), Real time obstacle avoidance for fast mobile robots, *IEEE Trans. Syst. Man Cybern.*, 19(**5**), pp1179-87

Chung, C.H. and K.S. Lee (1991), Neural network application to the obstacle avoidance path planning for CIM computer integrated manufacturing, *Proc. IROS '91. IEEE/RSJ Int. Workshop on Intelligent Robots and Systems '91, Intelligence for Mechanical Systems*, Osaka, Japan, **2**, pp824-8

DeJong, K., Learning with genetic algorithms : an overview, *Mach. Learning*, **3**, 2-3, pp121-38

Goldberg, D.E. and J.H. Holland (1988), Genetic algorithms and machine learning, *Machine Learning*, **3**, 2-3, pp95-9

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

Gough, N.E. et al (1994), Push-pull actuation mechanisms for robotic vision, *Proc. Melecon, Mediterranean Electrotechnical Conf.*, Antalya

Grefensette, J.J (1988)., Credit assignment in rule discovery systems based upon genetic algorithms, *Machine Learning*, **3**, 2-3, pp225-45

Griffiths,I.J. et al, Fuzzy-tuned Stochastic Scanpaths For AGV Vision, SICICA'97, to be published

Hughes, M. (1990), Improving products and processes- nature's way (genetic algorithms),*Ind. Man. and Data Systems*, **6**, pp22-25

Hulin,M. (1989), Logic partitioning: new optimisation methods compared, *Mikroelektronik fur die Informationstechnik*, 110, pp17-22

Kawakami et al (1991), Auto tuning of 3-D packing rules using genetic algorithms, *Proc. IROS '91, IEEE/RSJ Int. Workshop on Int. Robots and Systems '91. Intelligence for Mechanical Systems*, Osaka, Japan, **3**,pp1319-24

Khoogar,A.R. and J.K. Parker (1991), Obstacle avoidance of redundant manipulators using genetic algorithms, *IEEE Proc. of SOUTHEASTCON '91*, Williamsburg, VA, USA,**1**, pp317-20

Kimura, T. et al, (1993), Fuzzy path planning for an autonomous vehicle, *Japanese J. of Fuzzy theory and systems*, **5**, No. 4, pp627-636

Lee,P. and L.Wang.(1994), Collision avoidance by fuzzy logic control for automated guided vehicle navigation, *J. Rob. Syst.*, **11**, pp743-60

Lumbers,P.G. and A.S. Panda. (1994), Vision-based path following by using a neural network guidance system, *J. Rob. Syst.*, **11**, pp57-66

Michalewicz et al (1990), Genetic Algorithms and optimal control problems, *Proc. of the 29th IEEE Conf. on Dec. and Cont.*, Honolulu, **3**, 1664-6

Parker, J.K. et al (1989), Inverse kinematics of redundant robots using genetic algorithms, *Proc. 1989 IEEE Int. Conf. on Robotics and Automation*, Scottsdale, AZ, USA, **1**, pp271-6

Rao et al (1991), Optimal placement of actuators in actively controlled structures using genetic algorithms, *AIAA J.*, **6** ,pp942-943

Robertson,G.G and R.L. Riolo (1989), A tale of two classifier systems, *Machine. Learning.*,**3**,2-3, pp139-159

Schneider,K.(1989), Evolving the best solution, *Elect. Review*, 222, **19**, pp27-28

Suh,S.H. and K.G.Shin. (1988), A variation dynamic programming approach to robot path planning with a distance-safety criteria, *IEEE J. Rob. Autom.*,.**4**, No. 3, p334-349

Wang,L.X. and J.M. Mended (1992), Generating fuzzy rules from mathematical data with applications, *IEEE Trans. Syst. Man Cybern.*. **22**, No.6, pp1414-1427

Wang,T. et al (1996a), Kinematics models of autonomous vehicles and their applications, *Proc. of the Int. Soc. for Computers and their Applications 5th Int. Conf.* , USA

Wang,T. et al (1996b), A hybrid intelligent approach to navigation and control of AGV, *Proc. 11th Int. Conf. on Systems Engineering*, USA

Wang,T. et al (1996c), A human imitation controller for autonomous guided vehicles, *Proc. 12th Int. Conf. on CAD/CAM Robotics and Factories of The Future*, London, UK