BENCHMARKING ROBOT PATH PLANNING

ANDREW N. HAND, JAGRUTHI GODUGU, KAVEH ASHENAYI, THEODORE W. MANIKAS, ROGER L. WAINWRIGHT
University of Tulsa
Electrical Engineering
Tulsa, OK

ABSTRACT
The University of Tulsa, among other research institutions, performs research exploring the possibility of using Genetic Algorithms (GA's) to develop an algorithm for autonomous robot navigation. We have achieved significant success in this area. However, we encountered a problem. We realized there is no standard metric for comparing different algorithms developed by many researchers working in this field. Our most recent work involves developing a benchmarking system to evaluate different GA's. The concept behind our benchmark program is to provide a set of standard paths that have been mathematically evaluated and classified in different categories as Easy, Moderate, Difficult and Very Difficult.

INTRODUCTION

An autonomous robot performs without any external control from a human operator. The main components involved in the operation of an autonomous robot are visual detection of the environment, path planning and controls. Path planning has been defined as determining if a continuous and obstacle-avoidance sequence of positions and orientations of the robot exists, starting with the initial position and orientation of the robot, and ending with the goal position and orientation.

To date, several path-planning algorithms have been developed to achieve this end using several techniques from neural networks and fuzzy logic to genetic algorithms (Cazangi and Figuieredo 2002; Di Gesu et al. 2000; Hocaoglu and Sanderson 2001). However, there has been no standard for the comparison of the performance of these algorithms. Hence, the objective of our research is to define and formulate a metric for defining the complexity of an obstacle-filled navigation area, or *search space*, and use this metric to develop a mathematical rating system to classify the search spaces. This metric, the *complexity index*, provides a way to evaluate and classify search spaces. The aim is also to create a standard set of search space patterns with varying complexity index

values. This standard set of search spaces can be used as a means of evaluating the performance of different path planning algorithms.

SEARCH SPACE REPRESENTATION

Research in robot path planning considers the problem where a robot needs to travel from one point in a search space to another. A given area in the search space may contain several obstacles, which would obstruct the travel of a robot through that area. In order to provide a standard representation for search spaces, the area of interest is divided into a grid of cells (Fig. 1). Each cell of the grid occupies the smallest area into which the robot could fit. If the area covered by the cell contains an obstacle, it is marked as blocked; otherwise it is marked as clear. For the purposes of testing path planning, it is assumed the robot needs to travel from the upper-leftmost corner of the grid to the lower-rightmost corner. For this basic grid type, any number of algorithms may be used to plot a course for the robot to travel from start to finish.

THE BENCHMARK EQUATION

The variety of robot path solution algorithms has lead to a problem — how do we compare different algorithms to determine which one works best? The simple solution is to test the different programs on the same set of standard obstacle grids. However, there exists no standard set of common grids that every researcher can use. Often, the different researchers devise their own grids for testing their own algorithms. Thus we pose the question: how do we compare different algorithms if they were tested on different grids?

Our solution is to develop a set of benchmark test grids, and develop a grid complexity index, (CI), to quantify the complexity of these grids. This CI is a weighted linear sum of five terms (Eq. (1)). A through E are constant coefficients that weight the different terms (Table 1).

$$CI = A(\%Obstacle) + B\left(\#Obstacles\right) + C\left(Distribution\right) + D\left(TurnFactor\right) + E\left(Orientation\right) \quad (1)$$

The term “% Obstacle” is the ratio of the number of blocked cells divided by the total number of Cells, as expressed in Eq. (2). The “# of Obstacles” term counts the number of groups of cells that are joined together to form a single obstacle.

Term	Description	Coefficient
% Obstacle	How much of the grid is occupied	A
# of Obstacle	Number of obstacle groupings on the grid	B
Distribution	How well the obstacles are distributed on the grid	C
Turn Factor	How many turns the path takes	D
Orientation	Which types of movement can be used in the grid	E

Table 1 **Equation Terms and Coefficients**

$$\%Obstacles = \frac{\#OccupiedCells}{Total\,\#Cells} \tag{2}$$

The "Distribution" term uses Prim's Algorithm to compute the "spread" of the different obstacle groupings. Prim's Algorithm creates either a minimum or maximum spanning tree, depending on what is desired (see Fig. 2). In general terms, a spanning tree is a group of connections by which any node is connected to every other node, either directly or indirectly. The weight of the spanning tree is the sum of the distances involved in making the connections. Thus a Minimum-Weight Spanning Tree would connect all nodes together with the minimum possible total distance, and a Maximum-Weight Spanning Tree would connect the nodes for a maximum total distance. Our algorithm uses a type of normalization involving both the maximum weight and minimum weight trees (Eq. (3)).

$$Distribution = \frac{MaxWt}{MaxWt + MinWt} \tag{3}$$

To compute the "Turn Factor" term, random valid paths are generated through the grid. For each random path, the number of times the path takes a turn is added together. Figure 3 shows a path with three turns. Of all the paths, the paths with the greatest and least number of terms are used for the turn factor. The term is calculated as shown in Eq. (4).

$$TurnFactor = \frac{1}{MaxTurns - MinTurns} \tag{4}$$

Finally, the "Orientation" term is calculated by determining how the path must navigate the grid. There are three types of orientation considered: Row-Wise, Column-Wise, and Combinational (Hermanu et al. 2004; H-Sedighi et al. 2004). With Row-Wise movement, the path's row location is always increasing by one. For Column-Wise movement, the path's column location is always increasing by one. With Combinational Movement, either Column-Wise or Row-Wise may be used, and the path can switch

movement types as it progresses. Figure 4 shows examples of the different orientation types. We found that some "Orientation" types were more difficult to solve than others, and thus we assigned the different types certain complexity values, as shown in Table 2 (Godugu 2004).

Orientation	Term Value
Row-Wise	0.25
Column-Wise	0.5
Combinational, 1 or 2 Switching Points	1
Combinational, More than 2 Switching Points	2

Table 2 **Orientation Complexities**

SIMULATION RESULTS

A program with a graphical user interface (GUI) was developed (using Visual Basic) to allow for automatic calculation of CI for a given search space represented as a 10x10 grid cell. Also, a set of 10 sample search spaces were developed and classified (see table 3) using this program. The following values were used for the weights in Eq. (1): A=B=1, C=2, D=4 and E=8.

Using the three algorithms developed at the University of Tulsa (Geisler and Manikas 2002; Hermanu et al. 2004; H-Sedighi et al. 2004) we tested the accuracy of the CI. The three algorithms are progressively more capable, as verified by the test cases in Table 4.

Search Space	Category	CI	% Obst.	# of Obst	Turn Fact.	Obs. Dis.	Orientation
1	Easy	7.210	0.30	0.08	0.100	0.500	0.25
2	Easy	7.338	0.28	0.12	0.091	0.540	0.25
3	Moderate	10.266	0.25	1.00	0.100	0.817	0.25
4	Moderate	11.190	0.28	0.08	0.125	0.500	0.50
5	Moderate	12.225	0.30	0.32	0.100	0.696	0.50
6	Difficult	18.653	0.17	0.12	0.111	0.533	1.00
7	Difficult	21.269	0.45	0.20	1.000	0.730	0.50
8	Difficult	27.751	0.23	0.12	1.000	0.570	1.00
9	Very Diff.	43.471	0.47	0.16	1.000	0.670	2.00
10	Very Diff.	43.625	0.38	0.20	1.000	0.711	2.00

Table 3 **Complexity Index and Category Classifications**

While these results are promising, there is room for improvement. One factor that needs to be addressed is the way we calculate the distribution index. Currently, we use

Prim's algorithm, which measures distances between objects. However, if there is no difference in *relative* distances between obstacles when comparing different sized search spaces, the obstacle distribution will appear to be identical for both search spaces, which may affect the accuracy of the complexity index calculation.

Category	Search Space	Algorithm #1	Algorithm #2	Algorithm #3
Easy	1	X	X	X
	2	X	X	X
Moderate	3	X	X	X
	4		X	X
	5		X	X
Difficult	6			X
	7			X
	8			X
Very Difficult	9			
	10			

Table 4. **Algorithms Comparison**

We also need to improve the turn factor calculation. At present, we are using 50 randomly generated paths to calculate this metric. However, this may not be a good estimate if the random paths do not really represent the actual minimum and maximum number of turns required.

CONCLUSION

We have taken the first step in developing a benchmark that can be used to compare different robot navigation algorithms. Although it still requires further research, our complexity equation has given us a powerful tool for evaluating test grids. By conducting more research into improving the complexity index terms, we should be able to produce a more accurate metric that will be useful in classifying navigation spaces.

ACKNOWLEDGEMENT

We would like to acknowledge support of Cymstar and The University of Tulsa for this project.

REFERENCES

Cazangi, R. R., and Figueiredo, M., 2002, "Simultaneous Emergence of Conflicting Basic Behaviors and Their Coordination in an Evolutionary Autonomous Navigation System." Proc. 2002 IEEE Conf. on Evol. Comp. (CEC '02), pp. 466-471.

Di Gesu, V., Lenzitti, B., Lo Bosco, G., and Tegolo, D., 2000, "A Distributed Architecture for Autonomous Navigation of Robots." Proceedings Fifth IEEE International Workshop on Computer Architectures for Machine Perception, 2000, pp. 190 - 194.

Geisler, T., and Manikas, T. W., 2002, "Autonomous Robot Navigation System Using a Novel Value Encoded Genetic Algorithm." 45th IEEE Int. Midwest Symp. on Circuits and Systems, Tulsa, OK, pp. 45-48.

Godugu, J., 2004, "Development of a Benchmark for Robot Path Planning," Master's Thesis, University of Tulsa.

Hermanu, A., Manikas, T. W., Ashenayi, K., and Wainwright, R. L. 2004, "Autonomous Robot Navigation Using a Genetic Algorithm with an Efficient Genotype Structure." Intelligent Engineering Systems Through Artificial Neural Networks: Smart Engineering Systems Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Complex Systems and Artificial Life, C. H. Dagli, A. L. Buczak, D. L. Enke, M. J. Embrechts, and O. Ersoy, eds., ASME Press, New York, pp. 319-324.

Hocaoglu, C., and Sanderson, A. C. 2001, "Planning Multiple Paths with Evolutionary Speciation." IEEE Trans. Evolutionary Computation, Vol. 5, pp. 169-191.

H-Sedighi, K., Ashenayi, K., Manikas, T. W., Wainwright, R. L., and Tai, H.-M., 2004, "Autonomous Local Path Planning for a Mobile Robot Using a Genetic Algorithm." Proc. 2004 Congress on Evolutionary Computation (CEC2004), Portland, OR, pp. 1338-1345.

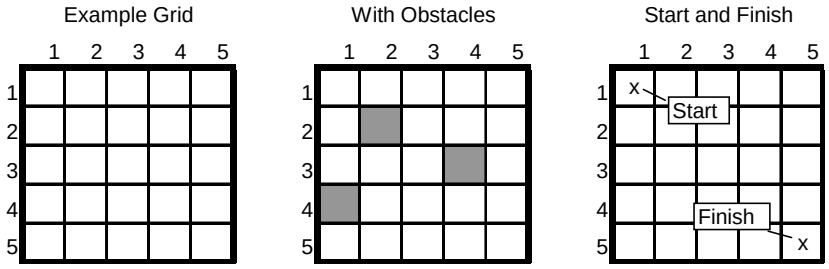

Figure 1 **Robot Navigation Grids**

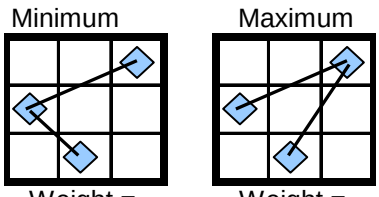

Figure 2 **Spanning Trees**

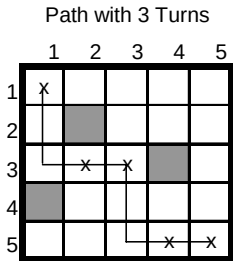

Figure 3 **Turn Factor**

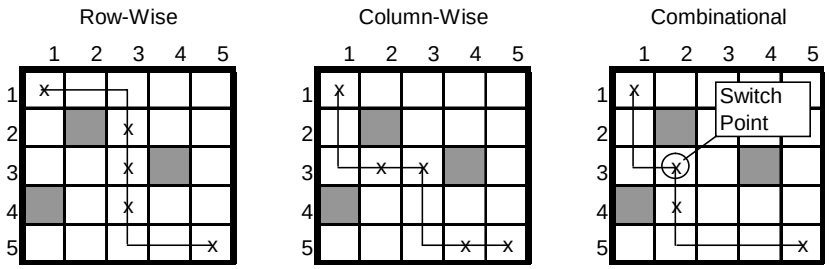

Figure 4 **Orientation Types**