A Potential Maze Solving Algorithm for a Micromouse Robot *

L. Wyard-Scott Q.-H. M. Meng †

ART (Advanced Robotics & Teleoperation) Lab
Department of Electrical Engineering
University of Alberta
Edmonton, AB, T6G 2G7, CANADA
E-mail: wyard@ee.ualberta.ca, mmeng@ee.ualberta.ca

***Abstract*—**
Discretely assigned potential levels can be effectively used in making autonomous route decisions for a mobile robot to reach a goal. This paper demonstrates methods of assigning and manipulating these artificial potentials to provide locally optimized path choices while maintaining the integrity of the potentials. The basic algorithm is improved by retaining information of the number of decisions that have been made. Results from implementation and simulation of the algorithm for a Micromouse maze-solving robot are presented. Consideration is given to implementation using a limited power microprocessor.

I. Introduction

This paper provides experimental and simulated results of a simple, yet effective, algorithm to solve maze-type problems based upon discretely assigned virtual potentials in a restricted environment.

The goal is to provide locally optimized routing choices based upon odometry and limited sensory data. The starting position of the automaton is known, as is the goal location, but knowledge of the region between the start and finish points is limited or unobtainable.

For this paper, the example of the well known Beam Robot Competition Micromouse Maze [1] is used. The maze consists of 16 by 16 squares of size 18cm by 18cm. The starting point is one of the 4 corners at which the mobile robot begins with a clockwise orientation; the goal location is one of the four squares which lie in the centre of the maze. To prevent graphical demonstrations from becoming cluttered, only one quadrant of the regulation size maze is used. One-quarter of a typical maze structure is shown in Fig. 1, to which the solving routine is applied. The regular nature of the corridor intersections and restricted movement in an x or y direction allows the algorithm to be constructed with great efficiency. This is of advantage when implementing the algorithm on a microprocessor which has limited computational power and memory size.

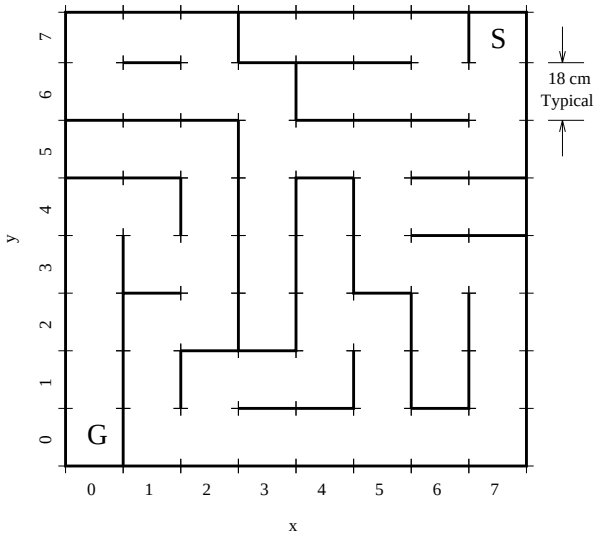

Figure 1: A typical one-quarter micromouse maze structure. S – autonomous robot starting position, G – goal location.

II. Potential Assignments

The assignment of potentials is the step in the maze solving algorithm that will determine the effectiveness with which the goal is obtained. The approach is to assign potentials in such a way that the initial assignment and manipulation of the potentials in the solving algorithm will not require extensive computational resources. Other potential methods [2] [3] are less suited for small-scale applications due to this constraint.

The simplest manner of assignment for an environment such as the Micromouse maze is to define the potentials at the centre of the grid squares. The potential value will be related to the distance required to travel

* This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant OGP170446 to M. Meng.

† To whom all correspondence shall be addressed.

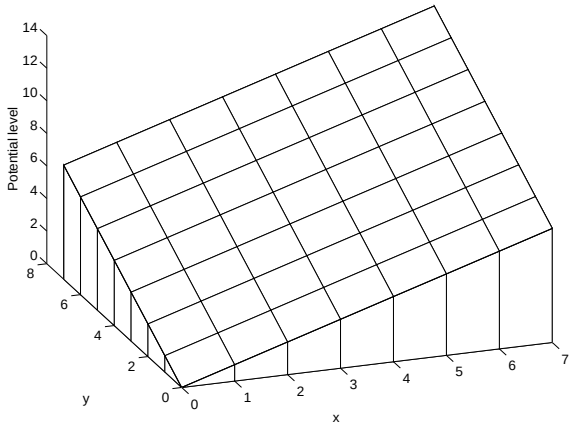

Figure 2: Assignment of initial potentials in a Cartesian environment. $V(x, y) = x + y$.

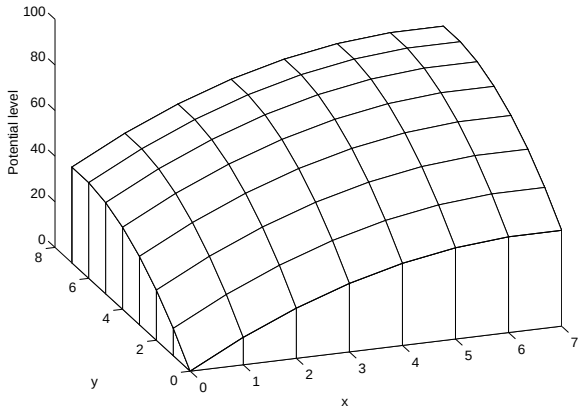

Figure 3: Initial potential distribution where examination of the problem yields that the highest probability of solution is in routes along the bounds of the terrain.

to the goal if no obstacles are present:

$$V(x, y) = \phi(\Delta x, \Delta y) \tag{1}$$

where $V(x, y)$ is the scalar potential value, and $\Delta x = x - x_g$ and $\Delta y = y - y_g$ are the Cartesian distances of the point of interest from the goal point. $\phi_x(\Delta x, \Delta y)$ is a discrete distribution function which could arise from the physical nature of the terrain, or from statistical analysis of solutions for similar problems.

If the goal coordinates are specified as the origin of the Cartesian coordinate frame, a potential distribution for the micromouse maze problem is

$$V(x, y) = K(x + y) \tag{2}$$

where K is a constant of proportionality which will be of interest when optimizing the solving algorithm for memory and computational constraints. An example of the distribution of Eq. (2) with $K = 1$ is shown in Fig. 2. This figure demonstrates the strong physical analogy that the potential method provides. If the mobile robot were to start anywhere (except at the goal) in an environment with no obstacles, then a 'rolling' action would occur toward the goal. If analysis of various mazes yields that the solution is typically along the sides of the maze, then the potential assignments can be adjusted to make use of this information. As an example,the distribution could be specified by

$$V(x, y) = 5(x + y) - (x - 4)^2 - (y - 4)^2 + 32 \tag{3}$$

Fig. 3 is a plot showing the potential distribution of Eq. (3), which would yield a better result than that of Fig. 2 under such circumstances.

III. Solving Algorithm

Assignment of potentials at discrete intervals in a manner that would result in an optimized goal obtaining action under the assumption that no obstacles exist is the basis for the maze solving algorithm. When obstacles do exist, a simple algorithm can be used to manipulate these potentials based upon odometry and simple obstacle detection sensory data such that a path is locally optimized. There are two criterion for such an algorithm:

- Manipulation of potentials must be done in a such a manner as to retain the integrity of the assigned potentials; and
- Local minima in the modified potentials must be avoided (or at least dealt with) such that the mobile robot will find a solution if one exists.

Sensory data is assumed to contain short range information about obstacles within one square unit of resolution of the potential assignments (for the Cartesian method) around the point which has been assigned the discrete potential. In the Micromouse example, the potentials are assigned to locations corresponding to intersection of the maze hallways. The obstacle sensory information that is available at any given intersection point is restricted to detection of walls to the North, South, East, and West. This information is stored along with the potentials in an array of size $(2x_{max}+1)$ by $(2y_{max}+1)$ where x_{max} and y_{max} are the Cartesian dimensions of the terrain. Allocation of the array in this manner is an exploitation of the nature of the maze. The values stored in the array between the potential values are the potential values of infinitely thin obstacles. A value of zero indicates that no wall is present, and a large value, V_{max}, is assigned if a wall is present. Fig. 4 demonstrates the structure of this array.

Local optimization of a path refers to the manner in which the decision about which direction to proceed is made. Based upon the sensory data and the potentials, the choice is made to proceed to the adjacent point

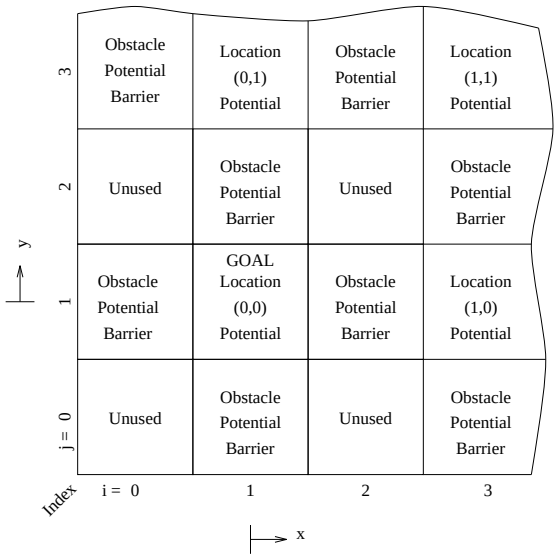

Figure 4: The structure of the array containing sensory data and the assigned potentials.

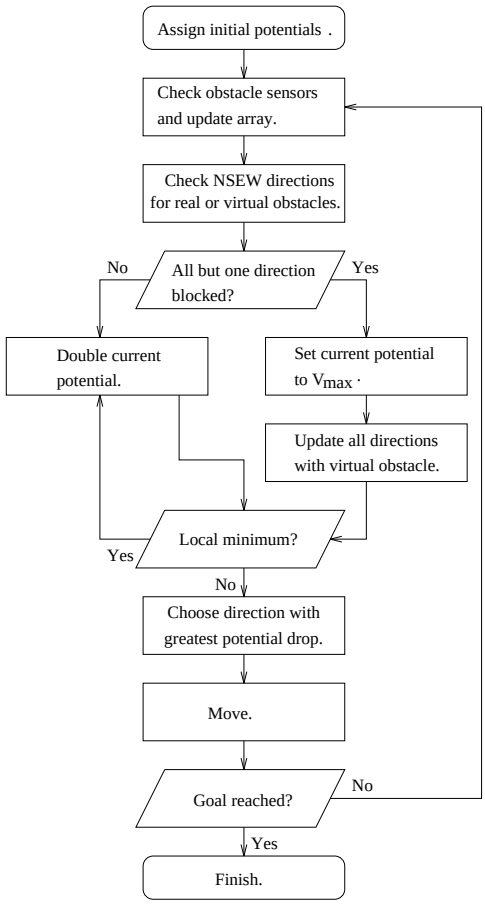

Figure 5: Flow chart of the maze solving algorithm.

which has the smallest potential value. Although the choice will yield the best results (depending upon the initial potential distribution), the decision is not based upon a complete picture of the obstacles between the start and the goal. The solution is therefore *locally* optimal. The algorithm is presented in flow-chart form in Fig. 5.

First, a sweep of the sensors is done to update the potential barriers posed by the obstacles. [1] If there is an obstacle which blocks passage, then the corresponding array element is assigned a value of V_{max}. A check of the array values is then performed. If a direction is either blocked by an obstacle or has a potential of V_{max} (a *virtual* barrier assigned through potential modification), then the direction is not considered. If all but one direction is blocked by a real or virtual barrier, then the current point can be effectively blocked off in memory; no access to this location will be required in the future. If there is more than one possible direction, then the current potential is doubled. This provides added 'potential energy' as there will be situations in which the obtaining the goal point will require 'climbing' the potential hill. As a safe-guard, the new potential is checked to see if it is a local minimum. This guarantees that a halt to the algorithm does not occur before a solution has been found. Now that a suitable model of the terrain has been generated, the direction to proceed is chosen based upon the potential levels of the adjacent locations that are not blocked by either a virtual or real obstacle. The direction which has the greatest potential drop with respect to the current potential is chosen. Movement then occurs in this direction to the adjacent location. The procedure is continued until the goal is reached.

[1] For the case of the Micromouse maze, the potential barrier is binary (the direction is blocked or not). The algorithm could be adapted to provide a measure of the *cost* to enter the adjacent region. [4]

IV. Implementation

The maze-solving algorithm was implemented on a mobile robot test-bed system which utilizes Motorola Corporation's 68HC11 microcontroller with 32K of external RAM. The physical structure of the mobile robot is shown in Fig. 6. Of the five infrared proximity detectors, three are used for detection of the presence of the walls. The range of these sensors is limited so that only walls around the current position can be detected. Differential steering is used to provide tight cornering which is required in the maze environment.

The algorithm itself was implemented in a version of C called Interactive-C. [5] Although the computational ability of the HC11 is respectable, optimization is still a good idea in order to leave system resources open for other tasks (navigation, motor control, etc.).

Optimization is achieved by selecting the potentials and maximum value such that it can be stored in a

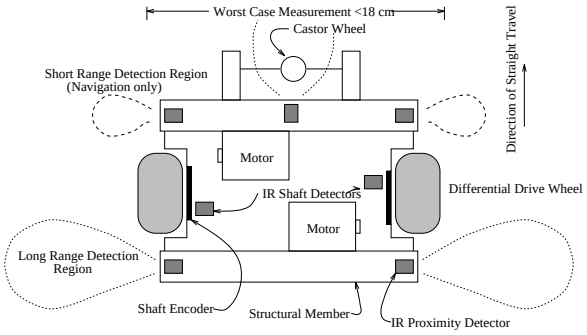

Figure 6: A diagram of the mobile robot test-bed.

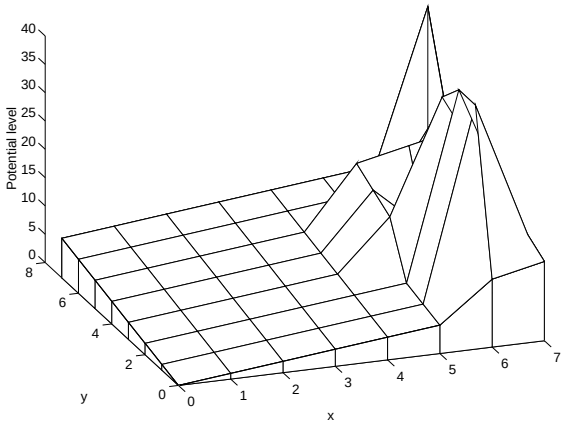

Figure 7: A snapshot of the potential levels after 17 moves. A potential of 40 (V_{max}) indicates that the location has been blocked off in memory, either because barriers are present all around, or the solution is not obtained through the point.

character array ($V_{max} = 255$). In this manner, not only is storage space minimized, but the resulting computational requirements are correspondingly reduced. The process of doubling the current position's potential (Fig. 5) can be implemented by a simple shift of the binary value. Additionally, comparisons will require fewer operations.

With this in mind, it is necessary to assign potentials so that the maximum value of the initial assignments is less than $\frac{1}{2}$ of the maximum value permitted by the storage length:

$$max(V(x,y)) = max(\phi(\Delta x, \Delta y)) < \frac{V_{max}}{2} \qquad (4)$$

This restriction rises from the fact that routes may have to be retraced, but no more than once.

A. Results

Performance of the algorithm in the robotic test-bed was exactly the same as that obtained through simulation.

Fig. 7 provides a picture of how the initial potential distribution of Eq. (2) with $K = 1$ has been modified after running the algorithm in the maze of Fig. 1. Fig. 8 shows the final potential distribution when the goal has been reached. V_{max} was selected as 40 to clarify the figures.

The path travelled by the automaton, shown in Fig. 9, indicates that there are a few situations in which a path is needlessly travelled more than once. This results from a situation where

$$V_{i+1} > 2V_{i-1} \qquad (5)$$

where V_{i+1} is the initial potential of any obtainable adjacent point further from the goal than the current location, and V_{i-1} is an adjacent point closer to the goal. In addition, multiple paths through loops are present if the potentials do not adhere to a similar condition, but where the potentials have already been modified by the algorithm. The solution to these problems could be found by assigning the initial potentials in a different manner. Instead another method can be used, and therefore the simplicity of the initial potential assignments is retained.

V. Improvements

A simple solution which prevents the problem of needless backtracking is to keep track of the number of route choices (decisions) made up to and including the current point. Storage of the value conveniently occurs in the unused elements of the maze array (shown in Fig. 4). The algorithm is then modified in two ways:

- The direction is chosen in such a way that regions which have not been explored (a 0 value in the array) have the highest priority; and
- Loop-backs are eliminated by comparing the current location's decision value with the value of the destination point. If the values are the same, then the path with this decision value is eliminated (by assigning a value of V_{max}).

Depending on the environment, the algorithm could be modified and improved in many ways.

A. Location of Potential Assignments

The distribution of the potentials for the algorithm used with the test-bed robot are regular in nature. Each potential is defined at a discrete point at the centre of a maze grid and odometry data and dead reckoning are used to move from one to another. The resolution of the assignments is inherent in the maze problem but could conceivably be increased to the resolution of the odometry data.

The nature of the terrain may allow a nonlinear positioning of the discrete potentials in some situations. For instance, the obstacles may be known to be large

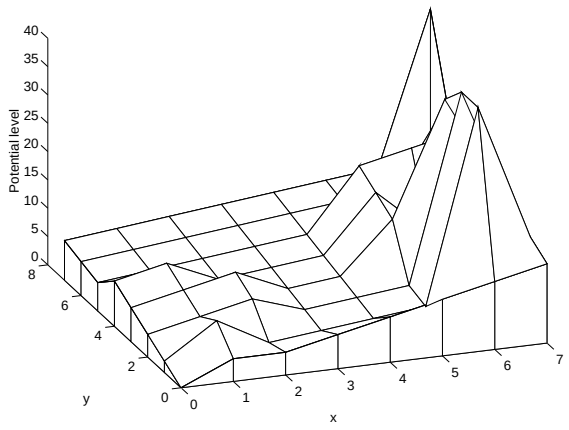

Figure 8: The final potential distribution after the goal has been reached.

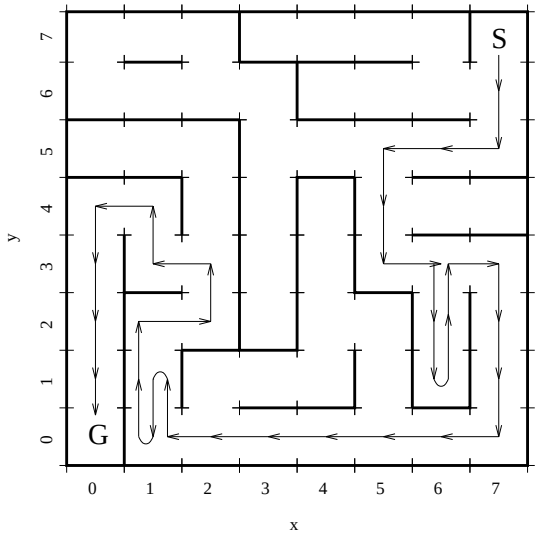

Figure 9: The route travelled by the mobile robot.

distances apart (allowing a rough path), but the goal point may be precisely located. Correspondingly, the resolution of assignments would be low far from the objective and higher as the goal is approached. Alternatively, a quadtree [4] representation could be used around obstacles, the advantage being smaller computational and storage requirements.

B. Time of Potential Assignments

In the example posed here, calculation of the potential distribution is the first step of the algorithm. Localization maps [6] [7] would allow division of the goal-obtaining task into small portions each solved independently and at different times. This would also allow for correction of odometry data, of particular importance particularly when the Cartesian limitation on movement is dropped.

Acknowledgement

L.W.-S. would like to thank Keith Lo for his input.

References

[1] "The Second International Beam Robot Olympics and Micromouse Competition: Event Rules and General Guidelines", Version 2.0, pages 53-57, Toronto, Ontario, Canada, 1992.

[2] Reid, M.B., Path Planning Using Optically Computed Potential Fields, *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2,pages 295-300, Atlanta, NC, 1993.

[3] Kim, J.-O. and Khosla, P., Real Time Avoidance Using Harmonic Potential Functions, *Proceedings IEEE International Conference on Robotics and Automation*, pages 790-796, Sacramento, California, 1991.

[4] Mitchell, J.S.B, An Algorithmic Approach to Some Problems in Terrain Navigation, *Artificial Intelligence*, 37, pages 171-201, 1988.

[5] Wright, A. and Martin, F., Chapter 6, "MIT 6.270 Course Notes", Interactive C Manual, pages 129-188, Cambridge, MA, 1993.

[6] Taylor, C.J. and Kriegman, D.J, Exploration Strategies for Mobile Robots, *Proceedings IEEE International Conference on Robotics and Automation*, vol. 2, pages 248-253, 1993.

[7] Curran, A. and Kyriakopoulos, K.J., Sensor Based Self-Localization for Wheeled Mobile Robots, *Proceedings IEEE International Conference on Robotics and Automation*, vol. 1, pages 8-13, 1993.