Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot

A. Zelinsky[1], R.A. Jarvis[2], J.C. Byrne[2] and S. Yuta[3]

[1]Intelligent Machine Behaviour Section, Electrotechical Laboratory, Tsukuba 305 Japan.

[2]Intelligent Robotics Research Centre. Monash University, Clayton 3168 Australia.

[3]Intelligent Robot Laboratory. Tsukuba University, Tsukuba 305 Japan.

Abstract- Much of the focus of the research effort in path planning for mobile robots has centred on the problem of finding a path from a start location to a goal location, while minimising one or more parameters such as length of path, energy consumption or journey time. A path of complete coverage is a planned path in which a robot sweeps all areas of free space in an environment in a systematic and efficient manner. Possible applications for paths of complete coverage include autonomous vacuum cleaners, lawn mowers, security robots, land mine detectors etc. This paper will present a solution to this problem based upon an extension to the distance transform path planning methodology. The solution has been implemented on the self-contained autonomous mobile robot called the Yamabico.

I. INTRODUCTION

The problem of planning a path of complete coverage of an environment by a mobile robot has not received significant research attention. Much of the focus of the research effort to date has centred on the problem of finding a path from a start location to a goal location, while minimising one or more parameters such as length of path, energy consumption or journey time. A mobile robot should be capable of planning other kinds of paths, such as have the capability to find paths which ensure the complete coverage of an environment. Possible applications of such paths include autonomous vacuum cleaners, lawn mowers, security robots, land mine detectors etc. Complete coverage paths can also allow a robot to systematically explore and map unknown terrains. This paper will presents a solution to the problem of complete coverage based upon an extension to the distance transform path planning methodology. Experimental results are presented by way of simulation and an implementation on the Yamabico mobile robot.

Using distance transforms for planning paths for mobile robot applications was first reported by members of our group in [1]. This approach considered the task of path planning to finding paths from the goal location back to the start location. The path planner propagates a distance wave front through all free space grid cells in the environment from the goal cell. The distance wave front flows around obstacles and eventually through all free space in the environment. For any starting point within the environment representing the initial position of the mobile robot, the shortest path to the goal is traced by walking down hill via the steepest descent path. If there is no downhill path, and the start cell is on a plateau then it can be concluded that no path exists from the start cell to the goal cell i.e. the goal is unreachable. Initially all the cells are initialised to high values. Fig. 1 shows an example of the distance transform.

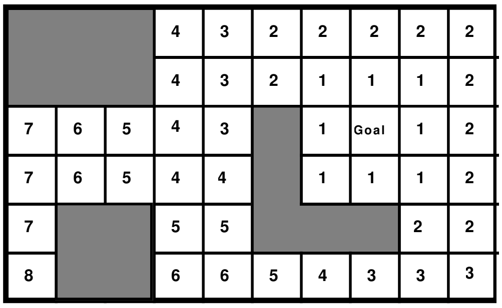

Fig. 1. Distance Transform Path Planning.

One significant advantage that distance transform path planning has over other path planning methods is that it can easily be induced to exhibit different types of robot navigation behaviours. Reference [1] described how the distance transform could be modified to produce "conservative", "adventurous" path planning behaviours in addition to the "optimum" i.e. shortest path behaviour. Experimental simulation results of the "complete coverage" path planning behaviour were reported [2]. However the algorithm was not published. The algorithm for complete coverage is as follows. To achieve the complete coverage path planning behaviour, instead of descending along the path of steepest descent to the goal, the robot follows the path of steepest ascent. In other words the robot moves away from the goal keeping track of the cells it has visited. The robot only moves into a grid cell which is closer to the goal if it has visited all the neighbouring cells which lie further away from the goal. Fig. 2 shows a "complete coverage" path example. This figure shows an environment with one obstacle, start (S) and goal (G) locations, values of the distance transform, and a "complete coverage" path from S to G.

The algorithm for the complete coverage of an environment is as follows:

```
Set Start Cell to Current Cell
Set all Cells to Not Visited
Loop
    Find unvisited Neighbouring cell with highest DT
    If No Neighbour Cell found then
        Mark as Visited and Stop at Goal
    If Neighbouring Cell DT <= Current Cell DT then
        Mark as Visited and Stop at Goal
    Set Current cell to Neigbouring cell
Loop End
```

Path of Complete Coverage Algorithm

13	12	11	10	9	8	7	7	7	7	7	7	7	7
13	12	11	10	9	8	7	6	6	6	6	6	6	6
S	12	11	10	9	8	7	6	5	5	5	5	5	5
									4	4	4	4	4
9	8	7	6	5	4	3	3	3	3	3	3	3	4
9	8	7	6	5	4	3	2	2	2	2	2	3	4
9	8	7	6	5	4	3	2	1	1	1	2	3	4
9	8	7	6	5	4	3	2	1	G	1	2	3	4
9	8	7	6	5	4	3	2	1	1	1	2	3	4
9	8	7	6	5	4	3	2	2	2	2	2	3	4

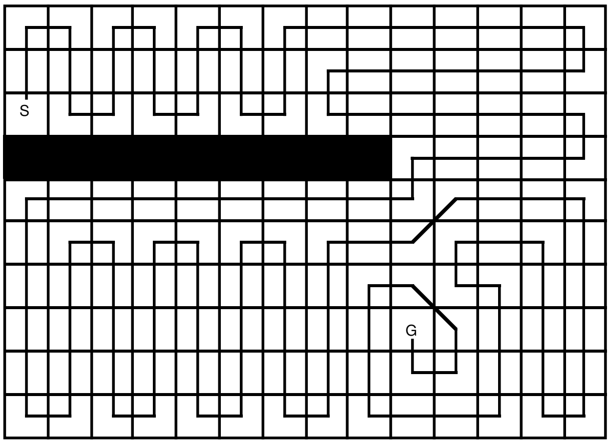

Fig. 2. Path of Complete Coverage

An advantage of this complete coverage strategy is that the start and goal can be specified. This is useful if for example, if a warehouse floor that needs to be cleaned. The robot can start at one end of the warehouse and end up at the other end, ready to enter the next warehouse. While this strategy does not guarantee the "complete coverage" path will be an optimum path i.e. the shortest possible and not unnecessarily visiting any cell more than once, the "complete coverage" produces a reasonable path with minimal secondary visits to grid cells. To find the optimum solution to this problem is equivalent to solving the Traveling Salesman's Problem. Each grid cell can be regarded as city with each city connected up to 8 other cities.

Closer observation of the above described path of complete coverage shows that the path produces too many turns. This is because the coverage path follows the "spiral" of the distance transform wave front that radiated from the goal. In certain configurations of obstacles in an environment this can produce unsatisfactory paths Complete coverage paths of the type shown in Fig. 2 are difficult to execute on a mobile robot that navigates by dead reckoning. The high number of turns and path segments will inevitably cause errors to be introduced in the estimation of the robot's correct position. Possibly for these reasons an implementation of complete coverage paths on an actual mobile robot has not been reported. A possible solution would be to use external navigation beacons. However this adds artificial structure to the environment. It is our aim to conduct experimental work in unstructured environments. In the next section a new approach is described which extends the work originally reported in [1]. The new approach generates a superior complete coverage path which has fewer turns and has longer individual straight path segments. This new approach facilitates its implementation on the self-contained Yamabico mobile robot. The Yamabico robot uses dead reckoning and the tracking of landmarks to maintain its position estimate while it is executing the complete coverage path.

II. THE APPROACH

The new approach to the problem of complete coverage is based upon an extension to the "path transform" that was developed by a member of our group [3]. The path transform was developed as an extension to the original distance transform. A disadvantage of the distance transform and most other path planning methods is that they only minimise the distance to a goal. However the safety of the robot is important particularly if the robot is operating in an unknown environment since there are uncertainties in the sensor data, i.e. the exact shape and position of obstacles. This problem is compounded by the uncertainty in the dynamic control of a robot i.e. the precise position of the robot is not always known by the robot's control system. Thus both minimum distance to a goal and safety of the robot need to be considered simultaneously during path planning. The path transform was successfully implemented on our Yamabico mobile robot [4]. Using the path transform the Yamabico was able to navigate along a known corridor and avoid unknown obstacles arranged in non-trivial configurations.

In the path transform approach, instead of propagating a distance from the goal wave front through free space, a new wave front is propagated which was a weighted sum of the distance from the goal together with a measure of the discomfort of moving too close to obstacles. This had the effect of producing a distance transform which has the desirable properties of potential fields [5] without suffering from the local minima problem. The path transform can be regarded as a numeric potential field.

The distance transform is extended to include safety from obstacles information in the following way. Firstly, the distance transform is inverted into an "obstacle transform" where the obstacle cells become the goals. The resulting transformation yields for each free cell in the data structure the minimal distance from the centre of the free space cell to the boundary of an obstacle cell. Refer to Fig. 3 for an example of the obstacle transform and the accompanying original distance transform.

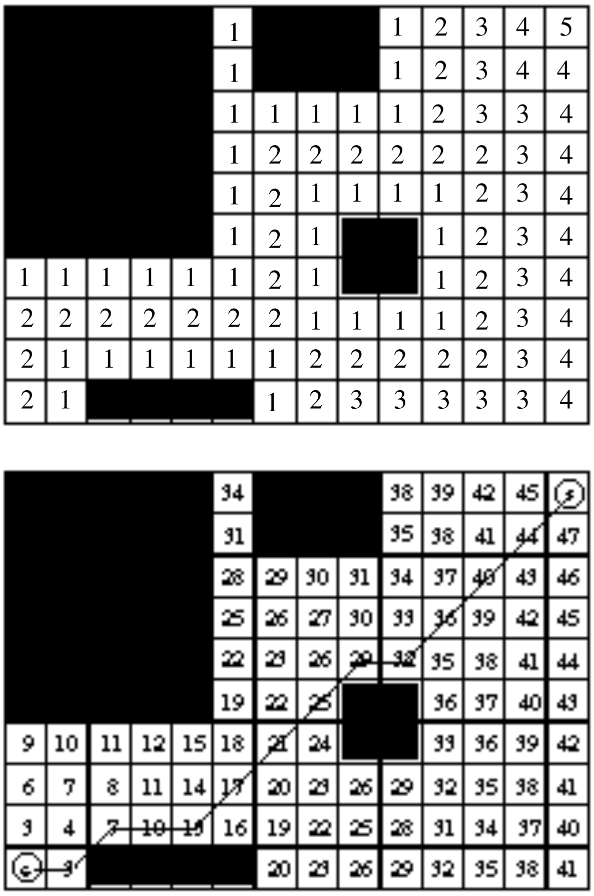

Fig. 3. Obstacle and Distance Transforms.

Finally, a second distance transform is generated through free space from the goal location using a new cost function. This cost function is referred to as the "path transform" (*PT*). The path transform for a cell c is defined as:

$$\mathrm{PT}(c) = \min_{p \,\in\, P} \left(\mathrm{length}(p) \; + \sum_{c_i \,\in\, p} \alpha \mathrm{obstacle}(c_i) \right)$$

where P is the set of all possible paths from the cell c to the goal, and $p \in P$ i.e. a single path to the goal. The function *length(p)* is the length of path p to the goal. The function *obstacle(c)* is a cost function generated using the values of the obstacle transform. It represents the degree of discomfort the nearest obstacle exerts on a cell c. The weight α is a constant ≥ 0 which determines by how strongly the path transform will avoid obstacles.

Finding the shortest path to a goal using path transforms is done in the same manner as finding the shortest path using distance transforms, i.e. by following the steepest path of descent. The path transform does not yield a transform with local minima, because all the costs of paths to the goal from each cell are calculated. The path transform for each cell is the minimum propagated path cost to the goal. Fig. 4 shows the path transform solution to the same problem posed in Fig. 3. This figure shows the result of applying two different obstacle clearance cost functions. The top figure shows a lighter cost function than the bottom figure. The degree of safety of the robot's path using the path transform can be "tuned".

A similar result to path transforms called "numeric potential fields" was reported by Barraquand and Latombe [6]. The numeric potential is computed in three (3) steps. Firstly, an "obstacle transform" is computed of the free space, from which a "distance skeleton" is extracted. Joining the highest values in the obstacle transform yields a distance skeleton. Secondly, the goal cell is connected to the distance skeleton and a distance transform is computed from the goal cell to all members cells of the distance skeleton. Thirdly, another distance transform is computed from the distance skeleton cells to all the remaining free space cells in the environment. This method is more complicated to compute than the path transform. Also, since it maximises clearance from obstacles it can deviate the solution path too far from the shortest path. Another drawback is this method does not consider clearance information. This method can guide the robot through narrow free space channels that are close to the goal thus endangering the robot. This problem is countered by removing channels that are narrow, but the completeness of solution is lost. The path transform does not suffer this drawback.

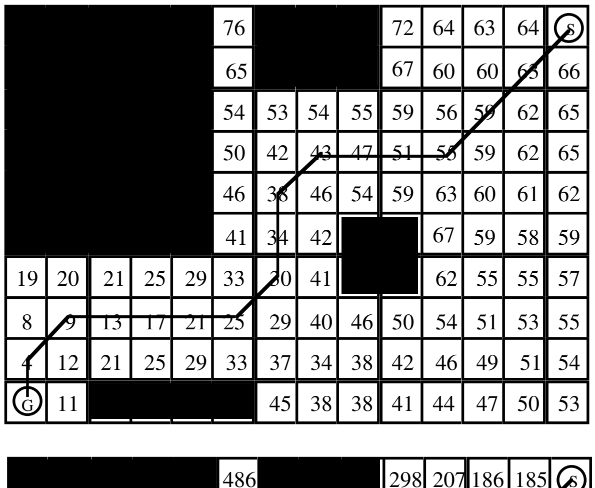

Fig. 4. Path Transforms with different cost functions.

The path transform forms a better contour path for a robot to implement a path of complete coverage than the contour path generated by the original distance transform. The distance transform forms circular contour patterns which radiate from the goal points. The path transform, on the other hand, forms contour patterns which slope towards the goal, but also follow the shape profile of obstacles in the environment. Fig. 5 and 6 show the distance and path transform solutions for the same planning problem. The path transform (Fig. 6) produces the better execution path, since it produces a path which has less turns and has more path segments that are straight. The distance transform (Fig. 5) produces a path which requires 37 turns and has an average length of 2.00 units for each path segment. In contrast the path transform in Fig. 6 produces a path which requires 19 turns and has an average length of 4.05 units for each path segment.

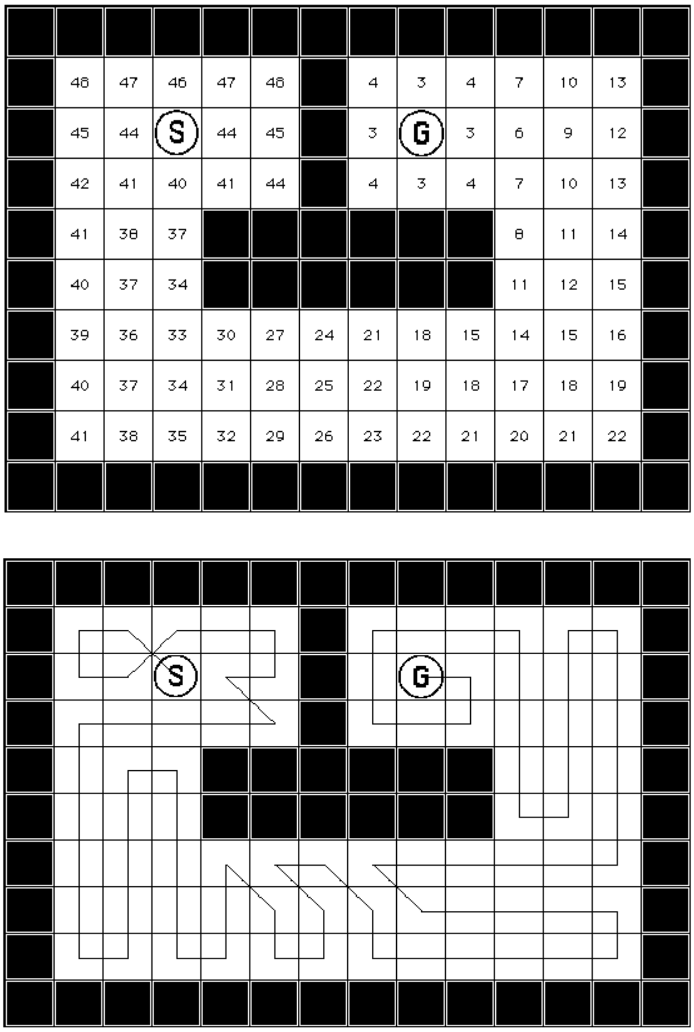

Fig. 5. Distance Transform Path of Complete coverage.

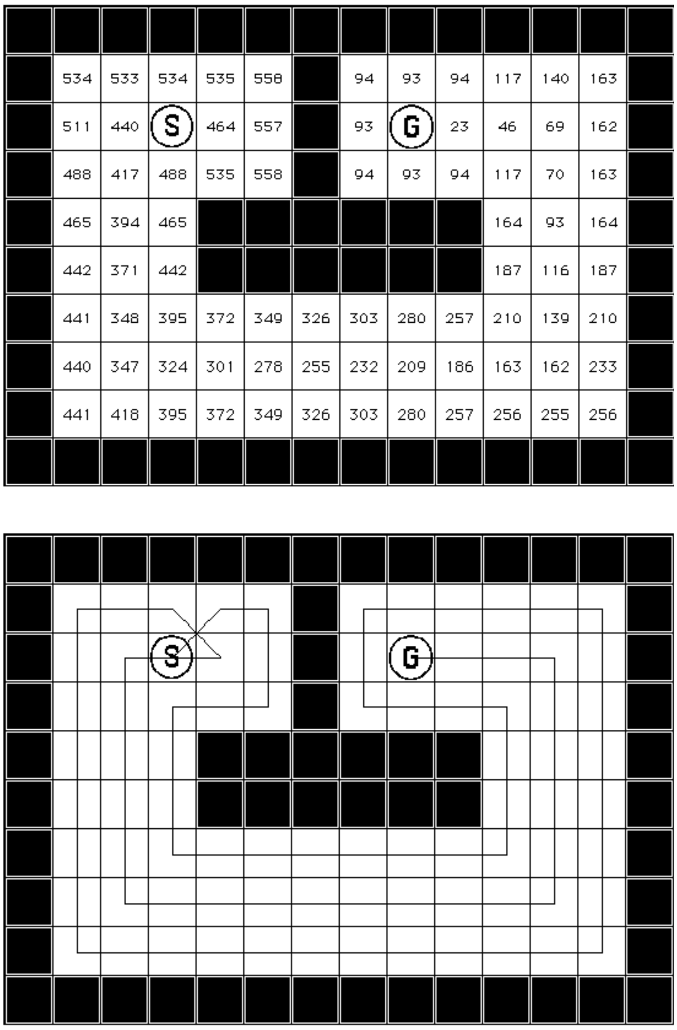

Fig. 6. Path Transform Path of Complete coverage.

III. Implementation

The aim of this research project was to implement the scheme of complete coverage described in the previous section on the Yamabico Robot, shown in Fig. 7. The Yamabico is an autonomous self contained mobile robot that was purpose built for the research of mobile robotics problems in indoor and outdoor environments [7]. The robot is equipped with two driving wheels mounted on a central axis and has optical and ultrasonic range sensors. Each hardware function of the robot is modularised to run on a single board computer. A master module implements decision making and coordinates the actions of each hardware module via shared memory and a communications bus. The master module is programmed to control the operations of the robot with the sensor based ROBOL/0 language developed for the Yamabico [8]. Software on the robot's master module runs under a multi tasking operating system (MOSRA) which was written specifically for the Yamabico.

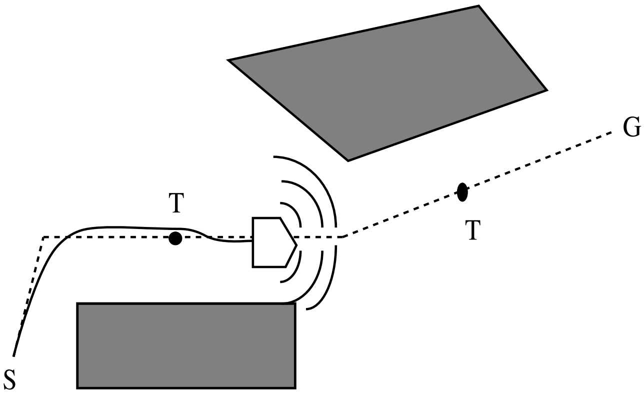

Fig. 8 Monitoring Path Execution by Tracking Landmarks

Fig. 7. The Yamabico Mobile Robot.

To implement the complete coverage scheme on the Yamabico some constraints must be placed on the approach described in the previous section. Namely, the grid used to generate the complete coverage path should be considered to be 4-connected instead of 8-connected. This constraint causes the planning scheme to generate a path that has longer straight path sections and turns are restricted to multiples of 90 degrees. This type of path is more suitable for implementation on a practical mobile robot like the Yamabico. The penalty for this constraint is that extra visits to a number of cells may result. Simulation studies have shown that the extra number of visits is small, less than 2% of the number of grid cells that represent the environment.

To correctly execute the planned path of complete coverage the robot must navigate with great accuracy. To rely on dead reckoning alone to execute a long and complicated path is unsatisfactory. Undoubtedly errors will creep into the estimate of the robot's position. Such errors are cumulative and will inevitably lead to an execution failure. The robot must periodically re-adjust its position estimate. This is done by sighting landmarks with the robot's ultrasonic sensors. The ultrasonic sensors measure the distances to the landmarks and these distances are compared with precomputed expected observation distances. Any errors that arise are and correcting dynamically during path execution. Refer to Fig. 8 for an illustration of position error correction. The precomputation of the landmark sensing points can be elegantly done by using data needed to compute the path transform. The path transform is computed by using a combination of the obstacle and distance transforms. The obstacle transform knows the exact distance that each free space grid cell is from the closest obstacle filled grid cell.

Since the robot is executing a path which closely follows the contours of obstacles in the environment, landmark tracking can be done in a straight forward manner. However, the obstacle transform does not contain all the information necessary to track the nearest landmark. The obstacle transform represents the distance to the closest landmark, but the direction of the nearest landmark is unknown. We modify the generation of the obstacle transform to include the directional information. This can be done in a straightforward manner. Distance transforms always record the minimum length path to a goal. At the time a new minimum value is recorded for a free space cell, we can easily record the direction of flow of the new distance minimum into current cell. Fig. 9 shows an example of a direction transform which was created at the same time as the obstacle transform. At path execution time the robot uses the direction and obstacle transforms together with its orientation to decide which ultrasonic sensor it should use to correct its estimated position. The Yamabico is equipped with front, back and side ultrasonic sensors. Therefore, the robot only uses the directions from the direction transform which are either N,S,E, or W to correct its position.

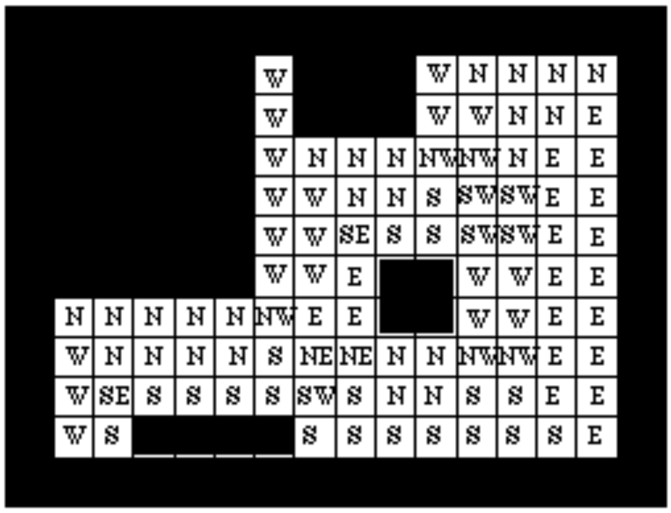

Fig. 9 Direction Transform for Tracking Landmarks

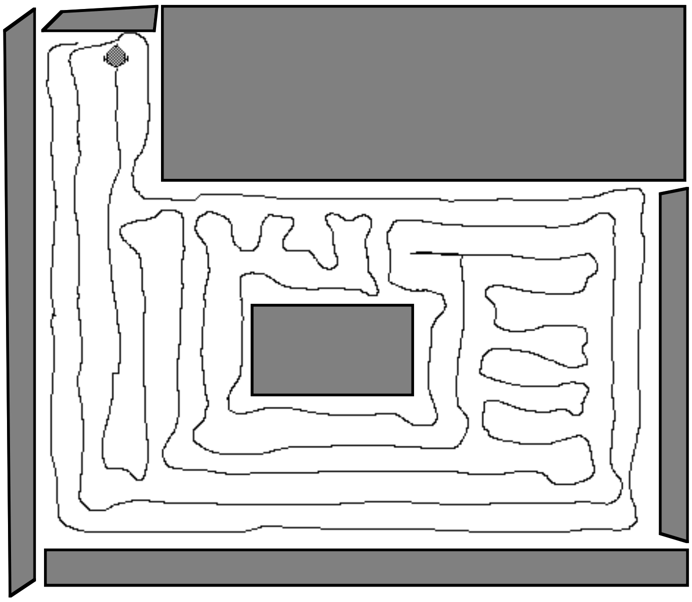

Fig. 10 Simulation Results

The proposed complete coverage scheme described above was initially implemented on the AMROS simulator[9]. This simulator models the kinematics of the locomotion system, and has realistic models of the robot's sensors. The simulator is programmed in the exactly same manner as the Yamabic robot. The simulator proved to be a valuable debugging tool. Software could be tested on the simulator before it was used on the robot. Figure 10 shows an example of output from the AMROS Simulator of the Yamabic navigating in a 7m x 6m room with a 1.5m x 1.0m obstacle in the room's centre.

Finally the complete coverage scheme was implemented and tested on the Yamabico. The Yamabico was successfully able to navigate in the environment depicted in the simulator.

IV FURTHER WORK AND CONCLUSIONS

The Yamabico was programmed to work in an known environment. If an unexpected obstacle is encountered along the planned path trajectory by the sensing systems, the Yamabico stops and waits for the obstacle to move. In future it is planned to add an obstacle avoidance procedure of the type described in [4]. In this procedure the robot uses its laser range finding system to determine the exact shape and location of the obstacle. A small local grid map is constructed and a path transform is computed. The path transform steers the robot from its current position around the obstacle to a goal point. The goal point is placed behind the obstacle on the previously planned complete coverage path trajectory. Once the robot has reached the goal point, it recommences the mission of complete coverage of the environment.

This paper presented a new complete coverage of an environment scheme based on a numeric potential field approach called the "path transform". It was argued that the path transform was an appropriate scheme for complete coverage, since this method delivered a path that could be readily executed by a robot. The new method incorporates the tracking of landmarks to ensure the correct execution of the planned path by the robot. An implementation of the scheme on the Yamabico robot was presented.

ACKNOWLEDGMENT

The authors are extremely grateful to Mr. Katsumi Kimoto for providing his simulator, which was an extremely valuable tool in debugging the robot programs. Special thanks are extended to Mr. Keiji Nagatani for his helpful assistance during the course of this research.

REFERENCES

[1] R.A. Jarvis and J.C. Byrne, "Robot Navigation: Touching, Seeing and Knowing", Proceedings of 1st Australian Conference on Artificial Intelligence, November 1986.

[2] R.A. Jarvis, J.C. Byrne and K. Ajay, "An Intelligent Autonomous Guided Vehicle: Localisation, Environment Modeling and Collision-Free Path Finding", Proceedings of 19th International Symposium on Industrial Robotics, November 6-10 1988, Sydney Australia.

[3] A. Zelinsky, "Environment Exploration and Path Planning Algorithms for a Mobile Robot using Sonar", Ph.D. dissertation, University of Wollongong, Department of Computer Science, October 1991.

[4] A. Zelinsky and S. Yuta, "Reactive Path Planning for a Mobile Robot using Numeric Potential Fields", Proceedings of 3rd International Conference on Intelligent Autonomous Systems IAS-3, February 15-18, 1993, Pittsburgh, USA.

[5] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", International Journal of Robotics Research, Vol. 5 No.1, pp90-98, 1986.

[6] J. Barraquand and J.-C. Latombe, "Robot Motion Planning: A Distributed Representation Approach", International Journal of Robotics Research Vol. 10 No.6, pp628-649, 1991.

[7] S. Yuta, S. Suzuki and S. IIda, "Implementation of a Small Size Experimental Self-Contained Autonomous Robot - Sensors, Vehicle Control, and Description of Sensor based Behaviour", Proceedings of 2nd International Symposium on Experimental Robotics, June 25-27, 1991, Toulouse, France.

[8] S. Suzuki, M.K. Habib, J. Iijima and S. Yuta, "How to Describe the Mobile Robot's - Sensor based Behaviour?", Journal of Robotics and Autonomous Systems, No.7, 1991, North-Holland, pp 227-237.

[9] K. Kimoto and S. Yuta, "A Simulator for Programming the Behavior of an Autonomous Sensor-Based Mobile Robot", Proceedings of International Conference on Intelligent Robots and Systems (IROS), July 7-10, 1992, Raleigh, USA.