Memory Based A-Star Algorithm for Path Planning of a Mobile Robot

Mahadevi S.[1], K. R. Shylaja[2], Ravinandan M. E.[3]

[1]M.Tech IV SEM, Dept. of CSE, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India

[2]Associate Professor, Dept. of CSE, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India

[3]Teminnova Technologies Private Limited, Bangalore, Karnataka, India

Abstract: ***Machine learning is a branch of artificial intelligence which concerns the construction and study of systems that can learn from data. Mobile robots are the one which have locomotion in the environment and are not fixed to any particular physical location. Planning path for these mobile robots is the most critical part, which can be accomplished using any of the well-known algorithms like Dijkstra's algorithm, Genetic algorithm etc., among which -- A-Star algorithm has been chosen currently for path planning in this paper. A-Star algorithm uses a best-first search and finds the least-cost path from a given initial node to one or many goal nodes. A- Star can be termed as a memory-less algorithm as it doesn't remember the path traversed between the same set of nodes initially. In this paper an approach to store and reuse of the pre-calculated paths has been proposed. Using this hybrid memory based A-Star algorithm mobile robots can be easily made to traverse the given environment. Initially the robots explore the environment using normal A-Star and once it learns about its environment, it exploits the environment by recalling the paths traversed and accomplishes the tasks even faster. Hence this algorithm is named Memory Based A-Star Algorithm.***

Keywords: Machine Learning, Path Planning, A-Star Algorithm, Mobile Robot, Bluetooth Communication

1. Introduction

1.1 What is machine learning?

Learning, like intelligence, covers a broad range of processes that it is difficult to define precisely. A dictionary definition includes phrases such as "to gain knowledge", or "understanding of", "or skill in", "by study", or "experience," and "modification of a behavioral tendency by experience". Here we focus on learning in machines. There are several parallels between animal and machine learning. Many models which have been developed to make machines learn have the roots from the way animals and humans learn. Machine learns whenever it changes its structure, program, or data (based on its inputs or in response to external information) in such a manner that it's expected future performance improves. Machine learning usually refers to the changes in systems that perform tasks associated with artificial intelligence (AI). Such tasks involve recognition, diagnosis, planning, robot control, prediction, etc. The changes might be either enhancements to already performing systems or synthesis of new systems.

Here we show a simple architecture of a typical AI agent in Figure 1. This agent perceives and models its environment and computes appropriate actions, perhaps by anticipating their effects. Changes made to any of the components shown in the figure might count as learning. Different learning mechanisms might be employed depending on which subsystem is being changed.

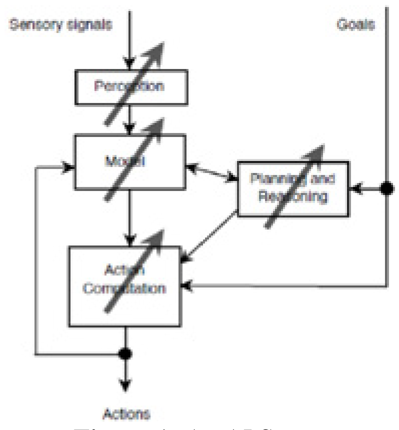


Figure 1: An AI System

1.2 Why learning is required for machines?

There are several reasons why machine learning is important. Some of them are:-

- Some tasks cannot be defined well except by example; that is, we might be able to specify input/output pairs but not a concise relationship between inputs and desired outputs. We would like machines to be able to adjust their internal structure to produce correct outputs for a large number of sample inputs and thus suitably constrain their input/output function to approximate the relationship implicit in the examples.
- Human designers often produce machines that do not work as well as desired in the environments in which they are used. In fact, certain characteristics of the working environment might not be completely known at design time. Machine learning methods can be used for on-the-job improvement of existing machine designs.
- Environments change over time. Machines that can adapt to a changing environment would reduce the need for constant redesign.

1.3 Path planning and its need in real-time environment

Path is a way or track laid down for walking which connects the start and end nodes. Path planning is a term used in robotics for the process of detailing a task into discrete motions. Planning a path in an optimal way is very critical and needs more attention.

Let's take a simple example from daily life to understand the importance of path planning and also to understand the essence of this paper. When we want to travel from place A to place B first we try to figure out are there any ways by which these
Two places can be connected. If we find many paths connecting these two places we choose one of the paths based on various criteria's like time of travel, distance of path etc., Next time when we need to travel between same pair of places then priority of choosing already used path will be more when compared to a new path.

The same concept is been proposed in this paper. Since the knowledge of choosing a path among many cannot come at once, it becomes a good example for machine learning. Here the robot updates its knowledge about the environment incrementally. Initially when robot starts exploring the environment it's like a new born child finally it grows to an expert by understanding its environment thoroughly.

We humans have master organ brain, which will decide the action in advance and instruct the supporting organs like hands, legs, eyes etc. to accomplish the planned action. In the same way planner plans the path for the robot and instructs A-Star algorithm to accomplish the task by leading robot to the destination.

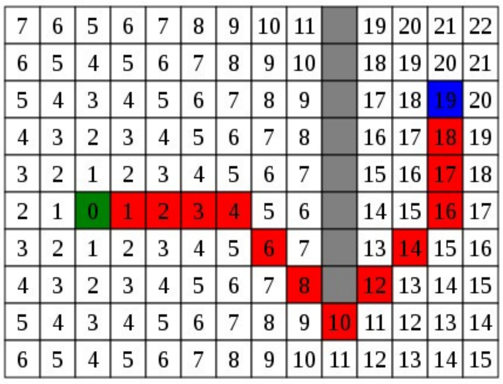
1.4 Existing system

In the existing system A-Star algorithm is used to find shortest path between start and end nodes. The A* algorithm combines features of uniform-cost search and pure heuristic search to efficiently compute optimal solutions. A* algorithm is a best-first search algorithm in which the cost associated with a node is **f(n) = g(n) + h(n)**, where g(n) is the cost of the path from the initial state to node n and h(n) is the heuristic estimate or the cost or a path from node n to a goal. Thus, f (n) estimates the lowest total cost of any solution path going through node n. At each point a node with lowest f value is chosen for expansion. Ties among nodes of equal f value should be broken in favor of nodes with lower h values. The algorithm terminates when a goal is chosen for expansion.

A* algorithm guides an optimal path to a goal if the heuristic function h (n) is admissible, meaning it never overestimates actual cost. For example, since airline distance never overestimates actual highway distance, and Manhattan distance never overestimates actual moves in the gliding tile.

Algorithm steps are as follows:
1. Create a search graph G, consisting solely of the start node, n_o. Put n_o on a list called OPEN.
2. Create a list called CLOSED that is initially empty.
3. If OPEN is empty, exit with failure.
4. Select the first node on OPEN, remove it from OPEN, and put it on CLOSED. Called this node n.
5. If n is a goal node, exit successfully with the solution obtained by tracing a path along the pointers from n to n_o in G. (The pointers define a search tree and are established in Step 7.)
6. Expand node n, generating the set M, of its successors that are not already ancestors of n in G. Install these members of M as successors of n in G.
7. Establish a pointer to n from each of those members of M that were not already in G (i.e., not already on either OPEN or CLOSED). Add these members of M to OPEN. For each member, m, of M that was already on OPEN or CLOSED, redirect its pointer to n if the best path to m found so far is through n.
8. For each member of M already on CLOSED, redirect the pointers of each of its descendants in G so that they point backward along the best paths found so far to these descendants.
9. Reorder the list OPEN in order of increasing f values. (Ties among minimal f values are resolved in favor of the deepest node in the search tree.)
10. Go to Step 3.

Figure 2 shows the simple example as how the shortest path is been calculated. Green tile represents start
Blue tile represents destination
Red tile represents path traversed
Grey tiles represent static obstacles



7	6	5	6	7	8	9	10	11	▨	19	20	21	22
6	5	4	5	6	7	8	9	10	▨	18	19	20	21
5	4	3	4	5	6	7	8	9	▨	17	18	🟦 19	20
4	3	2	3	4	5	6	7	8	▨	16	17	🟥 18	19
3	2	1	2	3	4	5	6	7	▨	15	16	🟥 17	18
2	1	🟩 0	🟥 1	🟥 2	🟥 3	🟥 4	5	6	▨	14	15	🟥 16	17
3	2	1	2	3	4	5	🟥 6	7	▨	13	🟥 14	15	16
4	3	2	3	4	5	6	7	🟥 8	▨	🟥 12	13	14	15
5	4	3	4	5	6	7	8	9	🟥 10	11	12	13	14
6	5	4	5	6	7	8	9	10	11	12	13	14	15

Figure 2: A-Star path finding example

1.5 Proposed System

In case of proposed system memory is given to the system which will store the paths which have been traversed already. It reuses the paths when needed, so this will reduce the effort of re-calculation of path by A-Star. Algorithm steps are as follows:
This algorithm works in two module, initial calculation module and recalling module.
1. Start the mission by inputting start and end nodes from the user.
2. Check whether there is a path between this pair of nodes using the recalling module.
3. If a path is present use that directly by sending it to robot.
4. If the path doesn't exists then use the initial calculation module which will instruct robot to follows the steps of normal A-Star algorithm to find the path connecting start and end nodes. 5.Store this path for future use by recalling module.

2. System Overview

2.1 System Architecture

The proposed architecture has been implemented using the following architecture. It consists of two main modules, module-1(Communication Module) and module-2(Memory Module), where module-2 in turn has two other modules Initial Calculation Module and Recalling Module. This is been tested on a mobile robot with following hardware specifications shown in table-1.

Table 1: Hardware specifications of ARM robot

Specification	*ARM robot*	
	Type	*Part name*
Primary Controller	ARM7(TDMI)	LPC2148
Secondary Controller	8051	89V51RD2
Communication	Bluetooth	HC-06
Obstacle detector	SONAR	Max EZ1
Motor driver	Dual H-bridge	L293D
Distance sensor	Infrared	slot sensor
Proximity sensor	Infrared	-
Battery	Lead acid	12V,1000mAH
Voltage regulator	linear	LM317

The robot which is shown in Figure 3, is built using all the specifications told in Table-1.

ARM robot is two wheeled robot driven by the DC motors at the rear end and a caster wheel at the front. Three obstacle sensor placed at the front, one being the distance sensor is placed at the center and two IR proximity sensor in right and left side of the bot. Both bots are built with aluminum chassis and the battery is placed below the chassis.

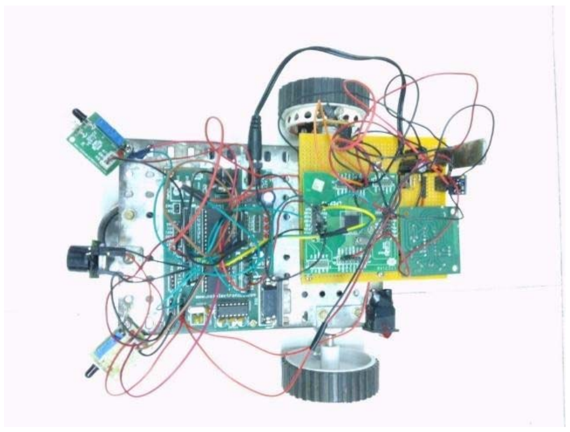
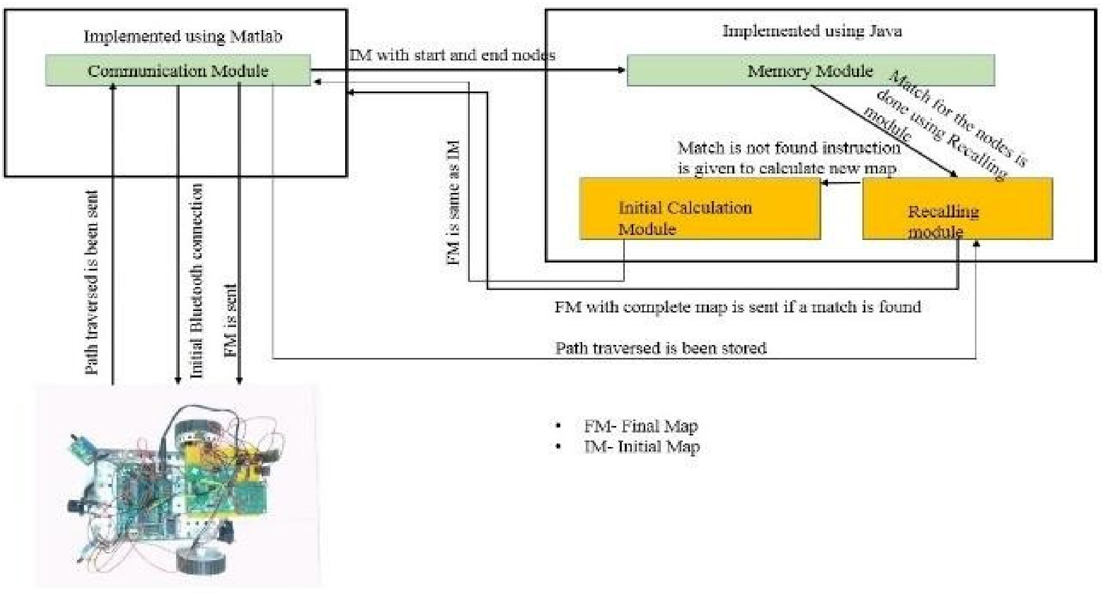


Figure 3: ARM robot top view

Figure 4 shows the complete system architecture. The working of the model is as follows. **Module-1** is currently being implemented in Matlab 2013a which mainly coordinates all activities of robot. This is named as Communication Module as this involves in exchanging of information to and from robot.


Implemented using Matlab
Communication Module
IM with start and end nodes
Implemented using Java
Memory Module
Match for the nodes is done using Recalling module
Match is not found instruction is given to calculate new map
Initial Calculation Module
Recalling module
FM is same as IM
FM with complete map is sent if a match is found
Path traversed is been stored
Path traversed is been sent
Initial Bluetooth connection
FM is sent
• FM- Final Map
• IM- Initial Map
Figure 4: Complete system architecture

Functions of Communication Module are:- Establish initial Bluetooth communication with the robot.

- Run the health monitoring system on the robot which will check whether all the components are working fine.
- If not a suitable error message is displayed.
- Next it takes inputs from users which give start and end nodes.
- This module constructs an initial map (IM) consisting of start and end nodes and forwards to the other module i.e. Memory Module.
- After processing the IM by the other module a final map (FM) is generated, which is then sent to robot via Bluetooth.
- If Memory Module sends back same initial map, this indicates that a new path need to be found.
- The same information is sent to robot and robot will use AStar algorithm which is burnt on its hardware to find new path.
- Once the robot reaches destination it sends back coordinates of the path through which it traversed.
- Finally, Communication module reads this and sends it to memory module.

Module-2 is been implemented using Java which acts as the memory part where the storage and retrieval of paths takes place. Hence it is named as Memory Module. This module doesn't have direct connection with the mobile robot, it only exchanges information with the Communication Module.

Functions of Memory Module: -

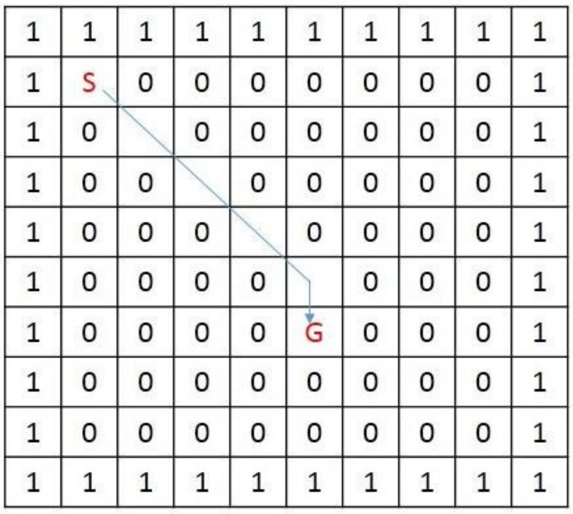
- The initial map (IM) which contains the start and end nodes is received from Communication Module.
- This module in turn has two other modules called Initial Calculation Module (ICM) and Recalling Module (RM).
- The combination of nodes is been searched by Memory Module by sending to RM, if a match is found then RM will retrieve the path(s) from its memory.
- If multiple paths are retrieved then best one can be chosen on our criteria, like shortest path among all or most recently traversed path etc.
- This path is used to construct final map (FM) which is then sent to Communication Module.
- If the match is not found then RM instructs ICM to plan new path.
- ICM sends back the same initial map to Communication Module.

3. Experiments and Results

The proposed idea of memory based A-Star Algorithm is been tested in following scenarios and the results obtained are as follows. In all the below figures which are used for explanation, S stands for Start, G stands for goal, 1 stands for obstacle and 0 stands for free cell.

3.1 Test with new start and end nodes:

In this case robot used its inbuilt A-Star algorithm to compute the path. Figure 5 shows the above scenario where S is start node with co-ordinates (1, 1) and G is goal with coordinates (6, 5). Information returned by the robot after traversing is (1, 1) (5, 5) and (6, 5). This is stored in RM for future use.

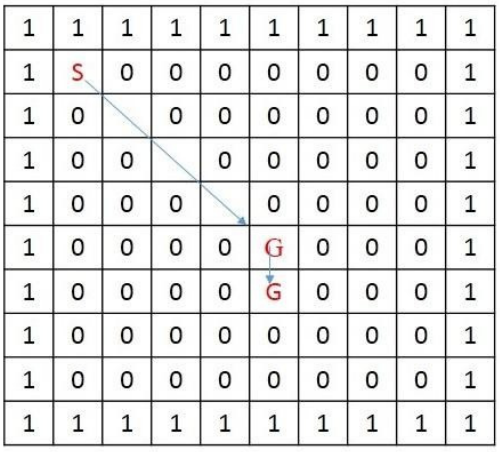


1	1	1	1	1	1	1	1	1	1
1	S	0	0	0	0	0	0	0	1
1	0		0	0	0	0	0	0	1
1	0	0		0	0	0	0	0	1
1	0	0	0		0	0	0	0	1
1	0	0	0	0		0	0	0	1
1	0	0	0	0	G	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1

Figure 5: Example showing initial path planning

3.2 Test with previously traversed start and end nodes:

In this case RM retrieved the path from its memory and was given to robot. Figure 6 shows this scenario. The same (1, 1) and (6, 5) were given as start and end nodes. RM returned the following path i.e. (1, 1) (5, 5), (6, 5). Now this is sent to robot where robot treats all nodes except the start node as destination nodes.

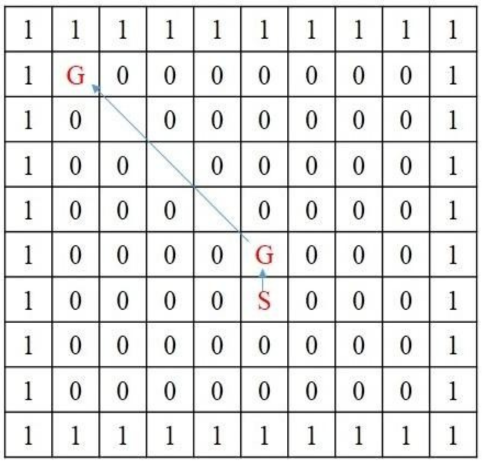


1	1	1	1	1	1	1	1	1	1
1	S	0	0	0	0	0	0	0	1
1	0		0	0	0	0	0	0	1
1	0	0		0	0	0	0	0	1
1	0	0	0		0	0	0	0	1
1	0	0	0	0	G	0	0	0	1
1	0	0	0	0	G	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1

Figure 6: Example showing retrieval of used path

3.3 Test with reversed nodes

In this case destination node of some previous trial is given as start node and vice versa. RM searches its memory and returns the relevant path in reverse order. Let's consider S as (6, 5) and G as (1, 1). RM returned the following path (6, 5), (5, 5), (1, 1). This scenario is shown in Figure 7.

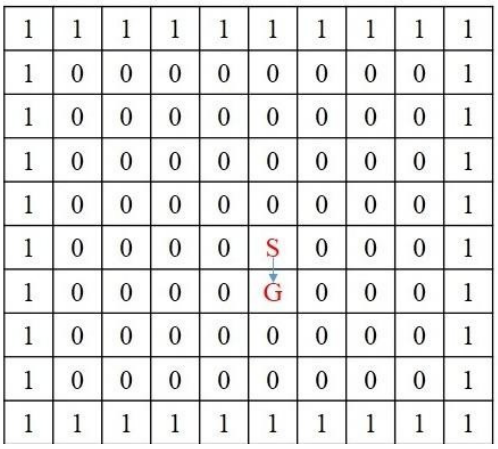


1	1	1	1	1	1	1	1	1	1
1	G	0	0	0	0	0	0	0	1
1	0		0	0	0	0	0	0	1
1	0	0		0	0	0	0	0	1
1	0	0	0		0	0	0	0	1
1	0	0	0	0	G	0	0	0	1
1	0	0	0	0	S	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1

Figure 7: Example showing retrieved reverse path

3.4 Test with intermediate Nodes

In this case there was no direct stored path for the given pair of nodes, but RM retrieves the path which contains these start and end nodes and generates map using this partial path alone. Let's consider S as (5, 5) and G as (6, 5). RM returned the following path (5, 5), (6, 5). This is shown in Figure 8.

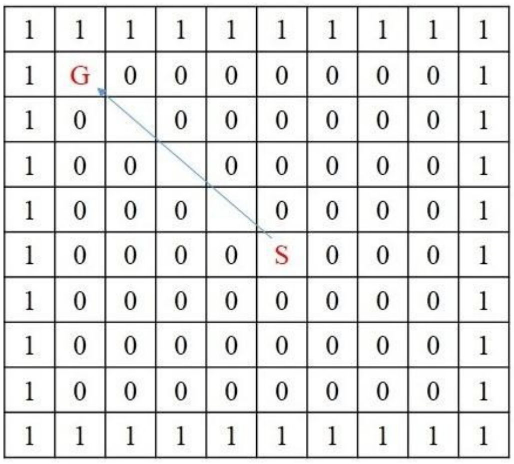


1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	S	0	0	0	1
1	0	0	0	0	G	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1

Figure 8: Example showing retrieved intermediate path

3.5 Test with Reversed Intermediate Nodes

This scenario is a mixture of 3 and 4scenarios. Let's consider S as (5, 5) and G as (1, 1). RM returns the following path (5, 5), (1, 1).



1	1	1	1	1	1	1	1	1	1
1	G	0	0	0	0	0	0	0	1
1	0		0	0	0	0	0	0	1
1	0	0		0	0	0	0	0	1
1	0	0	0		0	0	0	0	1
1	0	0	0	0	S	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1

Figure 9: Example showing retrieved reversed intermediate path

4. Conclusion

This paper mainly focuses on providing memory to memory-less A-Star algorithm. This will effectively reduce the task of recalculation of path between same set of nodes again and again. This eventually makes the robot to learn about its environment incrementally.

5. Future Enhancements

Here storing and retrieval of path is been proposed only for A-Star but this can be done using any of the path planning algorithms because Memory Module is completely a different module from the planning module.

References

[1] M. Batalin, G. Sukhatme, and M. Hattig. Mobile robot navigation using a sensor network. In Proceedings of IEEE ICRA, 2004.

[2] INTRODUCTION TO MACHINE LEARNING, Nils J. Nilsson Robotics Laboratory Department of Computer Science Stanford University,Stanford, CA 94305

[3] Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohamad Z, "A Simple Local Path Planning Algorithm for Autonomous Mobile Robots," International Journal of Systems ApplicationsEngineering& Development, Issue 2, Volume 5, 2011.

[4] HasanMujtaba, Gopal P Sinha," The State of Art on Navigational Algorithm for Path Optimization of a Mobile Robot," International Journal of Scientific & Engineering Research, Volume 4, Issue 9, September2013 22 ISSN 2229-5518.

[5] J. Su and W. Xie, "Motion planning and coordination for robot systems based on representation space," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 41, no. 1, pp. 248–259, Feb.2011.

Author Profile

Mahadevi S. received B.E. degree from Information Science and Engineering branch, Sapthagiri College of Engineering, Banglore in 2012 and currently pursuing MTech degree in Computer Science and Engineering from Dr. Ambedkar Institute of Technology, Bangalore in 2012-2014. Having a strong passion towards Machine Learning and Robotics.

Mrs. K.R. Shylajareceived MTech degree in Computer Science from Visvesvaraya Technological University. She is currently purusing her Ph.D from JNTU, Kakinada Andhra Pradesh, India. She has 14 years of experience as lecturer and Asst. Professor, Currently she is an Associate Professor, Department of Computer Science & Engineering, Dr. Ambedkar Institute of Technology, Bangalore.

Ravinandan M E received B.E degree in Computer Science Engineer from Bangalore University, MTech degree in Computer Science & Engineering from VTU. He had served as a scientist in DRDO and GE Healthcare. Currently he is MD & CEO of Teminnova Technologies Private Limited and also a Founder Member of Organization for Rare Diseases India.