ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

A Grid Based Robot Navigation by Using Priority Algorithm

Lalit Gehlod[1], Vaibhav Jain[2], Mala Dutta[3], Devesh Kumar Lal[4]

Asst.Professor, Computer Engg. Department, IET, DAVV, Indore, India [1]

Asst.Professor, Computer Engg. Department, IET, DAVV, Indore, India [2]

Asst.Professor, Computer Engg. Department, IET, DAVV, Indore, India[3]

Student, Computer Engg. Department, IET, DAVV, Indore, India[4]

Abstract: The searching of a block over grid is easier when the rows and columns i.e. m* n of a grid is fixed. But when the grid is dynamic or changes over time than in such situation we require a generalized algorithm for traversing over a grid. In these paper we develop an approach for searching an object and also able to avoid an obstacle which was placed in a junction (meeting point of row and column). Here, we use different algorithms like Dijkistra's, Best first search and A star algorithms. We develop an approach to find the block with minimum shortest path with the help of priority based algorithm.

Keywords: Grid solver robot, shortest path algorithm, Line follower robot, path planning, Grid based navigation, obstacle avoidance.

I. INTRODUCTION

The grid based robot navigation system is one of the most dynamic areas of material handling today. Transportation of raw materials and finished products is typical in an industry. Controlled transportation and product identification, as well as safe movement throughout the process, are the key to such type of installations [5]. The major problem that the robot faces while traversing over the grid is of path planning and identification of its co-ordinate we will discuss it in later section[1][2]. Firstly we will discuss the basic term related with these algorithm.

A grid is represent as the [m * n] matrix where m is the number of rows and n is the number of columns. The rows and columns may be black line which is drawn over a white surface or white line which is drawn on black surface. The robot which follows a single line is known as line follower robot the line is either black or white. Here we use IR sensors for sensing the line [8][16]. IR sensors are able to distinguish between white and black lines. IR sensors consist of a IR emitter and IR receiver pair. The high precision IR receiver always detects a IR signal. The white and black colors has different wavelength which can be compares by IR sensor [17]. To follow a Black line we require minimum two sensors. The sensors are placed one after another. To follow a line we requires two different condition i.e. when the left most sensors are in black line we have to take slow right turn and when the right sensors are in black line we have to take slow left turn[9]. This is the minimum condition to make the robot on line.

Similarly to traverse over the grid it follows the same condition but when the junction has been detected (when all the sensors are attain into the line it is detected as a junction) at such circumstances we have to take the decision whether to take left or right turn[5][6]. To navigate over a grid we have to follow Cartesian coordinate system for finding the current location on the grid. The robot set its initial location as (0,0) and maps all the quadrant according to it. The left node as (-1,0) the right node as (1,0)and the node below the origin is taken as(0,-1) and above the origin as (0,1) respectively. The robot has also to maintain its direction while moving forward, left or right it has to update the direction according to the turn. Updating of direction according to the TURN and its priority are explained in detail in later section [7][16]. The arduous problem in the field of robot navigation is when it consist an obstacle and we have to avoid such obstacle and search for the object. We have to search the object with the minimum searching algorithm. The obstacle and object is placed on the junction. The obstacle is a cubical block with half portion is colored with white and another half with black and we consider the purely black or white as an object [3][4].

To deal with Grid Based Robot Navigation problem we have to juxtapose with dump bot and real life problem by dealing with the searching an object which is kept at certain location in a house [6]. The primary step is to search for object in the same room and then the consecutive room. Similarly the algorithm by priority is follows such behavior.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

II. ENVIRONMENTS USED FOR ROBOT NAVIGATION

A. Type of Grid

We can use the grid of any dimension of [m * n] as shown in fig 1.The grid may also consist of combination of multiple grid. For such situation we have to place two extra sensors in below the center of both the wheel. These sensors are capable of to keep the robot into the grid with a condition i.e. when all the sensors in (white or black) we have to take (right or left) turn until the line is detected [5].

B. Grid Mapping

The grid is map according to Cartesian coordinate system. Robot sets the initial node as an origin and maps the entire grid according to origin and its direction which shows in fig 1.

C. Figures

1. Coordinate System of Grid

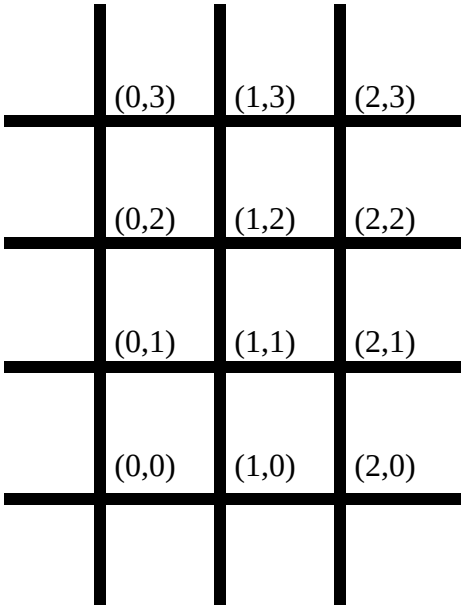

Fig 1 Grid [3*4](shows about a black line which is drawn in a white surface or vice-versa).The junction is labelled as according to its coordinate system.

2. Obstacle

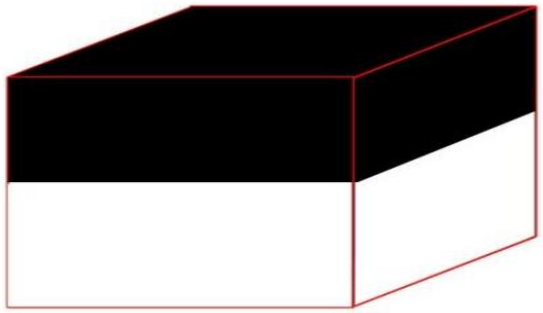

Fig 2 Obstacle (Two IR sensors are placed in front of the robot which is which is used to detect the two different colours are treated as an obstacle).

3. Object

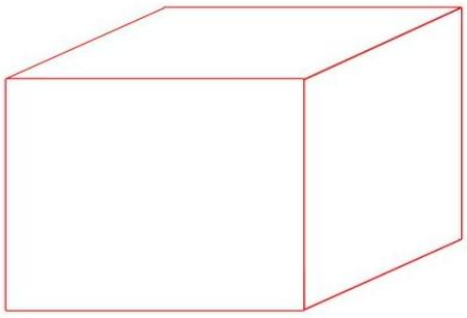

Fig 3 Object (The pure white and black block are consider as an object i.e. if both the sensors give the same value it is an object).

III. ALGORITHM USED

A. Dijkistra's Algorithm

As according to dijkistra algorithm we have to traverse the node closer to the origin which is set by the robot [11].

B. Best-First Search

According to best first search we have to traverse the node which is closer to the object. With the help of heuristic function [14].

C. A Star Algorithm

It is the combination of both dijkistra' and best-first search. Here we have to traverse a node which is closer to origin as well as closer to object [15].

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

IV. ALGORITHM WITHOUT OBSTACLE

A. Calculating Minimum Hops

For calculating minimum hops of a grid by formula is

M.H. =|x1 - x2| + | y1 – y2|

B. Calculating Direction

Table 1.

Direction	Turn	Set Direction	Set Co-ord.
North	Forward	North	Y++
North	Right	East	X++
North	Left	West	X- -
East	Forward	East	X++
East	Right	South	Y- -
East	Left	North	Y++
West	Forward	West	X- -
West	Right	North	Y++
West	Left	South	Y- -
South	Forward	South	Y- -
South	Right	West	X- -
South	Left	East	X++

C. Sensor Placing

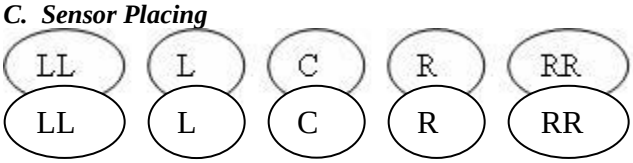

Fig 4 IR Sensor Placing (LL-Left Most sensor, L- Left sensor, C-Center sensors, R-Right sensors, RR-Right Most sensor)

C. Attributes used in Algorithm

I) Traverse stack

The value of (x, y) is added into traverse stack when the robot is encounter a junction on a grid. It also placed the value into the stack when an obstacle is detected.

II) Obstacle stack

When robot detect an obstacle on a grid the value(x, y) on that instance are place into the stack.

III) Priority

It maintains the priority of all the connected nodes. It contains the field (x, y, ptr) ptr represent the priority which is incremented by one every time the robot navigate (ptr: = ptr+1).These stack is responsible for taking the decision for the next node movement of robot .The robot check the max priority value in the stack on every junction to move in that direction corresponding value of (x, y) to ptr. After traversing the node the priority is reset to 0. If value of the priority of more than two coordinate are same than the next node is taken according to the direction. We will see at section IV (direction).

IV) Direction

When there is ambiguity in priority stack the next node is determine in order to N>E>S>W. The direction is given a weight N, E, S and W as 1,2,3,4 respectively. Required Numbers of direction possibilities are shown in section IV (B) table1.According to the use of such tables the direction is updated on every junction. Direction is also used in path planning concept which is discussed at section IV (E).

V) Next node

In every junction we have to find the next node which is done with the help of priority and direction as discuss in earlier section.

VI) Count

The variable ctx is works as an counter it keep the record that the number of junction the robot traverse over a grid it increment the value of ctx by one (ctx: =ctx+1).

VII) Current node

These variables are responsible for locating the current location in the grid. On every motion or turn the direction and current location is updated.

VIII) Detect junction

When all the sensors give a high value a junction is encounters.

E. Path Planning

In robot navigation over a grid the robot in any instances has to map its current coordinate in any junction. The robot start with direction north and co-ordinate(0,0) update the values of (x,y) according to turn as shown in section IV(B) table 1.These section is able to update its co-ordinate and direction. For an example if robot is an location (2,3) south direction and bot has to move the co-ordinate (0,1)We have to calculate the minimum hop is required to traverse between these two location by the M.H.(minimum hop) formula in section IV(A) is M.H.= |2-0|+|3-1|= 4.So we get 4 min step to move in location(0,1).we have equate (2,3)-(0,1)and comparing with the direction the next node is generated as (2,2)while view in the table of south direction it gives by reducing the y- - by moving forward[13]. again by comparing (2,2)-(0,1) next node is (2,1) by getting with forward movement as y - -.after detecting the junction at (2,1)and comparing with (0,1) the next node has been detected as(1,1) means x++ which has been given by taking right turn and direction changes to west .and again the last comparison between (1,1)-(0,1) we see in table of west direction and by moving forward as X- - its gives the co-ordinate (0,1) which is the shortest route from (3,2)[12]. But in the case of obstacle the process is similar but if obstacle is detected we have to make entry in traverse stack

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

and obstacle stack. The bot has to backtrack and again find the next node. After encounter a junction next node has to be compare whether the co-ordinate has been traverse earlier or not and it may contains an obstacle. Such co-ordinate has been given the priority 0 in the priority stack [11].

F. Flow Chart without obstacle

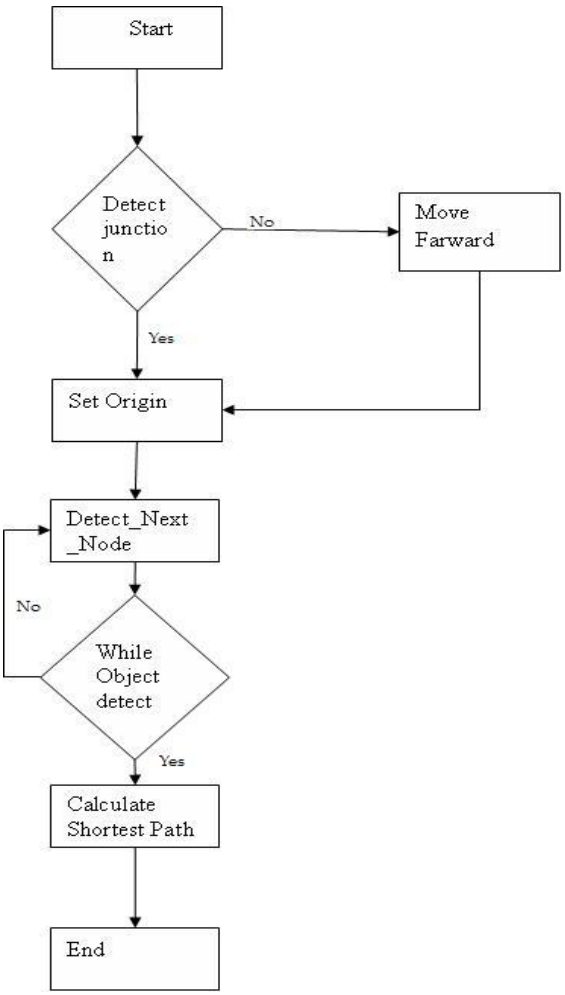

Fig 5 Flow Chart without Obstacle.

G. Algorithm (Without Obstacle)

1 start
2 Ctx:=0;
3 Int x:=0,y:=0;
4 Object detection:
5 If s1===high && s2==high || s1==low&& s2==low
6 Return 0
7 Else
8 Return 1
9 End if
10 Detect junction:
11 **If** ll=high&& l==high&& c==high&& r==high&& rr==high
12 Ctx:=Ctx+1;
13 End if
14 Traverse stack(int x , int y)
15 **/***Stores the value of traversed coordinate*/
16 Set priority(int x , int y)
17 /* set value 1 to all the connected node */
18 Direction()
19 /* it is implemented as table 1*/
20 detect junction
21 Set x:=0 ,y:=0
22 **While**(object detection!=0)
23 Traverse stack(x,y)
24 Set priority(x,y, value)
25 Detect next node
26 Check max priority in priority stack
27 **If** priority is max
28 Take turn
29 Change direction
30 Else
31 Check in Direction turn.
32 End if
33 End while

V. ALGORITHM WITH OBSTACLES AND OBJECTS

A. Arrangement over a Grid

The grid consists of any m*n matrix where an two obstacles and object is placed over a junction of(x1, y1), (x2, y2), (x3, y3) respectively. After locating the initial position the bot traverse the node which is closer to origin. We can also design an alternate motion of a bot like it has to move in

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

clock wise direction as -f-r-r-f-f-r-f-r-f- after completing such turns bot will move a one complete cycle. The cycle is enhancing by introducing one forward turn in every left or right turn. The approaches is failed in the situation when the grid is not continues or an obstacle is present in the route.

B. Flow Charts with Obstacle

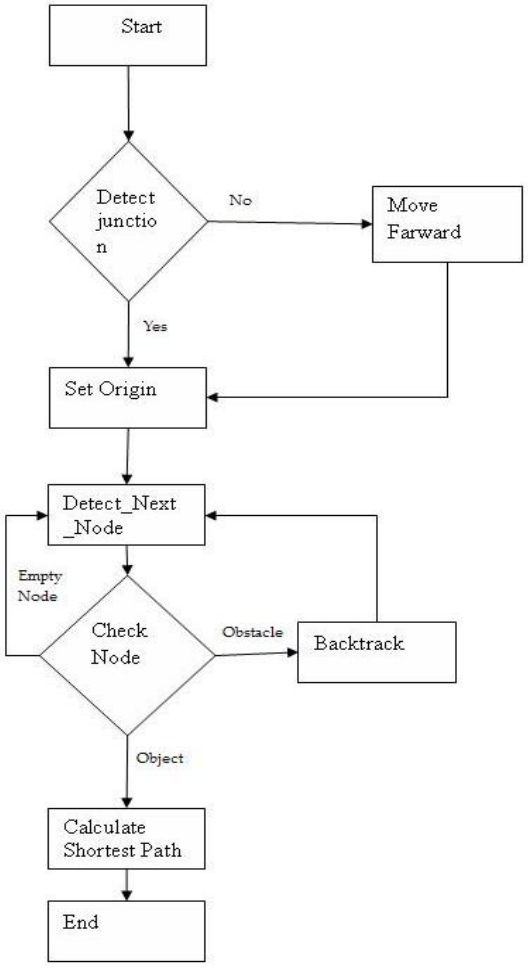

Fig 6 Flow chart With Obstacle.

C. Detailed Algorithm (With Obstacles and Object)

```
1   start
2   Ctx: =0;
3   Int x: =0, y: =0; /*Global Variable */
4   Char Dec;
5   Object detection:
6   If s1==high && s2==high || s1==low&& s2==low
7   Dec=true;
8   Return  True
9   Else if s1==high && s2==low||s1==low&&s2==high
10  Dec=false
11  Return false
12  Else
13  dec=not;
14  //do nothing
15  End if
16  Detect junction:
17  If ll==high&& l==high&& c==high&& r==high&& rr==high
18  Ctx:=Ctx+1;
19  End if
20  Traverse stack(int x,int y)
21  /*Stores the value of traversed coordinate*/
22  Set priority(int x,int y)
23  /* Stores the values of connected nodes and its priority*/
24  Write stack(int x ,int y)
25  Detect object()
26  //s1 & s2 are the two front sensors
27  If s1==high&& s2==high || s1==low&&s2==low
28  Return object
29  Else
30  Return obstacles
31  End-if
32  Forward()
33  /*condition for forward */
34  Left turn()
35  /*condition for Left turn */
36  Right turn()
37  /*condition for Right turn */
38  Path planning(int x1 ,int y1)
39  /* Select destination node by priority stack*/
40  Int x2,y2;
41  //cal M.H.
42   Take turn according to direction
43  detect junction
44  Set x:=0 ,y:=0
45  While(Object detection !=true)
46  If Dec==false
47  Go to obstacle;
48  Begin:
49  Traverse stack(x,y)
50  Set priority(x,y ,value)
51  /* set value 1 to all the connected node */
52  Detect next node
53  Compare with traverse stack
54  If node present
55  Set priority 0
56  Else
57  //compares with priority stack
58  Priority stack()
59  End if
60  Check max priority in priority stack
61  If priority is max
62  Set x:= ,Y:= //values of x and y are of max priority
```

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 8, August 2013

63 Path planning(x,y)
64 Else
65 Check in Direction turn.
66 End While
67 If dec=true
68 /* Calculate shortest distance by M.H. and path planning*/
69 Obstacle:
70 Traverse stack(x,y)
71 Backtrack ( )/*move back until junction is detected */
72 Detect next node();
73 Go to Begin;

VI. RESULT

This algorithm is implemented on 4×4 matrix where bot encounters 16 junctions. The architecture of robot is consisted two stepper motor, arduino ATmega 8 microcontroller, one LCD, one motor driver IC and six IR sensors. Five sensors are placed on bottom of the bot for detecting the line. The most corner sensors keep the record of the junction while three sensors in the middle keep the bot in line. We use one sensor in front for detecting an object.

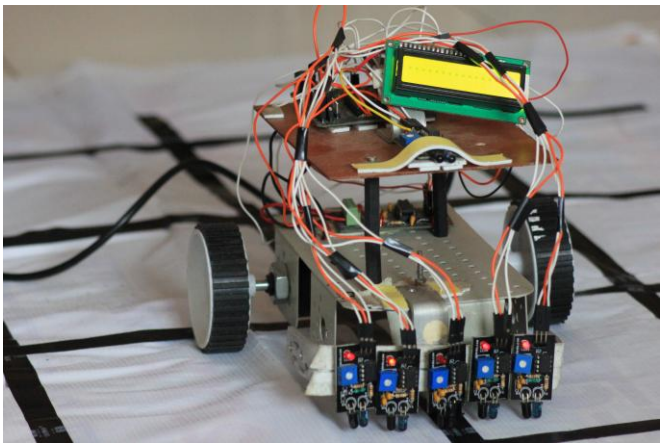

Fig 7 Robot used for testing this algorithm.

We kept an object on junction (2,2) and placed the bot on initial node i.e.(0,0). The bot navigates according to priority based algorithm is 00->01->11->10->20->21->22.

VII. CONCONCLUSION

These paper will be useful in the field of robot navigation system .We have discuss the algorithm for searching a desired block with shortest route. We introduce the concept of priority of the node while traversing. We also applied the obstacle avoidance technique with the help of stacks. We develop two different algorithms where overall robot navigation is simple and errorless. This paper represents path planning and navigation over a grid with a single bot. In future work, we can introduce multiple bot over a grid which will suppose to communicate with each other about its self-position and the obstacle that has been placed over a grid. Therefore, it is easier for the other bot to know about obstacles and object location on the grid so that it will become easier to make the shortest route by using this value.

REFERENCES

[1] Sebastian Thrun and Arno Bucken, "Learning Maps For Indoor Mobile Robot Navigation", April 1996, CMU-CS-96-121, School of Computer Science, Carnegie Mellon University Pittsburgh, A 15213.
[2] Sreekanth Reddy Kallem , Department of computer science, AMR Institute of Technology, Adilabad,JNTU, Hyderabad, A.P, India "Artificial Intelligence Algorithms" IOSR Journal Of Computer Engineering (IOSRJCE) ISSN: 2278-0661, ISBN: 2278-8727 Volume 6, Issue 3 (Sep-Oct. 2012).
[3] Joachim Hertzberg and Frank Kirchner. Landmark-based autonomous navigation in sewerage pipes. In Proceedings of the First Euromicro Workshop on Advanced Mobile Robots (EUROMICRO '96), pages 68{73. IEEE Computer Society Press, 1996.
[4] Hans P. Moravec. Sensor fusion in certainty grids for mobile robots. AI Magazine, pages 61{74, Summer 1988).
[5] Michael J.Milford Janet Wiles Gordon F.Wyeth, "Solving Navigational Uncertainty Using Grid Cells On Robots," November 2010, school of engineering systems, Queensland University of Technology, Brisbane, Australia, journal.pcbi.1000995.
[6] Carsten Buschmann Florian Muller and Stefan Fischer, "Grid-Based Navigation for Autonomous, Mobile Robots ",Institute of Operating Systems And Networks, Technical University of Braunschweig Braunschweig, Germany.
[7] Sebastian Thrun,"Robotic Mapping: A Survey",Feb 2002,CMU-CS-02-111,School of Computer Science, Carnegie Mellon University Pittsburgh.
[8] 8-bit AVR Microcontroller with 16k bytes in System Programmable Flash Brief Details By ATMEL http://www.atmel.com/images/doc2466.pdf.
[9] Robert J.Szczerba Danny Z.Chen John J. Uhran,"A Grid-Based Approach For Finding Conditional Shortest Paths In An Unknown Environment ", Nov 1994,Department of computer Science and Engineering University of Notre Dame ,Notre Dame, Indiana 46556,U.S.A.
[10] Websites on algorithm by amit" http://theory.stanford.edu/~amitp/GameProgramming/Heuristics.html".
[11] Book on Algorithm by Thomas H. Cormen Charles E.Leiserson Ronald L. Rivest and Clifford Stein, "introduction to algorithm" Second edition.
[12] Reference book on Data Structure by Mark Allen Weiss, "Data Structure and Algorithm Analysis in C++"third edition.
[13] Algorithm Concept by Narasimha Karumanchi"Data Structures and Algorithms Made Easy" second edition.
[14] Websites on best first search is http://www.artint.info/slides/ch03/lect3.pdf.
[15] Websites on A Star algorithm is http://www.informatics.sussex.ac.uk/courses/FP/AstarAlgorithm/AstarAlgorithm4.pdf.
[16] Ahmedullah Aziz, Md. Shafayat Hossain and Mohammad Wahidur Rahman "Programming and construction of Ahmedullah-A Fast Grid solver robot" International Journal of Information Technology, Control and Automation (IJITCA) Vol.3, No.1.
[17] IR Sensors reference from Robosoft System "IR Sensor Based Obstacle Detection Sensor Module".
[18] Richard T.Vannoy, "Design a Line Maze Solving Robot",April2009http://www.pololu.com/file/0J195/line-maze-algorithm.pdf.