KSHITIJ 2016
THE TECHNO-MANAGEMENT FEST
21st- 24th JAN

EMBETRONIX

CHALLENGING QUEST IN EMBEDDED ELECTRONICS

IIT KHARAGPUR

Embetronix

Abstract— Navigation is a major challenge for autonomous, robots. The problem can basically be divided into positioning and path planning. In this tutorial we present an approach which we call grid-based navigation. Though we also propose a scheme for path finding, we focus on positioning. Our approach uses minimal environmental infrastructure and five IR LED pairs on the device itself. Starting out from a predefined tile and orientation in the grid, the robot can autonomously head for destination tiles in the grid. On its way it determines the current location in the grid using a finite state machine by picking up line crossing line crossing events with its sensors.

INTRODUCTION

A key ability needed by an autonomous, robot is the possibility to navigate through the space. The problem can basically be decomposed into positioning and path planning. Though we also propose a scheme for the latter, we clearly focus on the detection of the current position by monitoring grid crossing events by using two light sensors. Especially if the robot is severely resource-constrained, simple schemes are favorable to elaborated algorithms. Rather simple sensors and actuators as well as a limited computing platform also demand simple, robust techniques due to inaccuracy and the lack of resources. The paper is concluded by a summary and an outlook to future work.

In this tutorial we would be dealing with how the grid follower robot works along with its algorithm. A grid follower is basically a line follower with a modified code which helps it in traversing a grid. Just as a simple line follower, a grid follower takes inputs from various sensors to detect its position on the grid.

ENVIRONMENT

The robot navigates on a grid (see Figure 1) which regularly divides the ground into square tiles that are identified with Cartesian coordinates. As mentioned earlier, the robot starts out from a predefined position (e.g. the Centre of tile 0,0) and orientation (e.g. North, which means looking up the y-axis). It is sufficient that the full grid is rectangular and all tiles are the same size. It is not required that the robot knows the dimension (n, m) of the grid. It is satisfactory to only request navigation to tiles that really exist.

Heading from tile to tile, the robot distinguishes eight driving directions: north, north-east, east, south-east, south and so on.

Depending on the size of the robot and the sensitivity of the luminance sensors, different tile sizes and contrasts between plane and grid are possible.

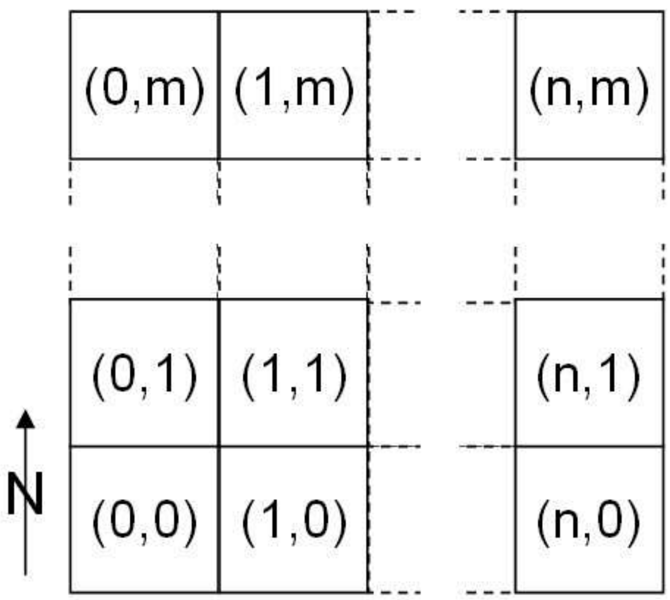
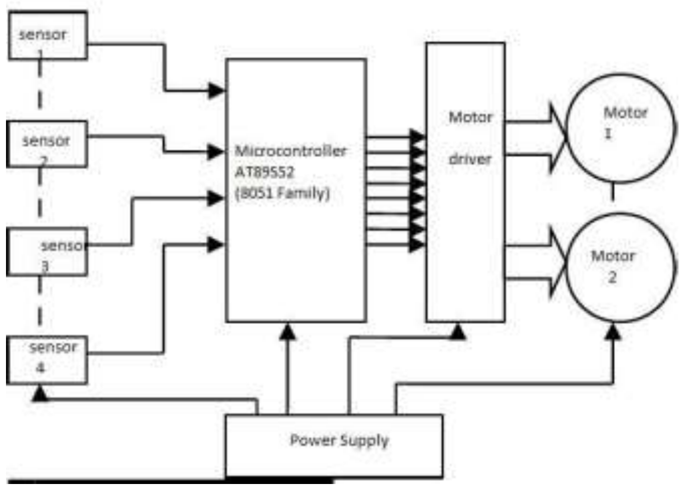


Figure 1: The grid

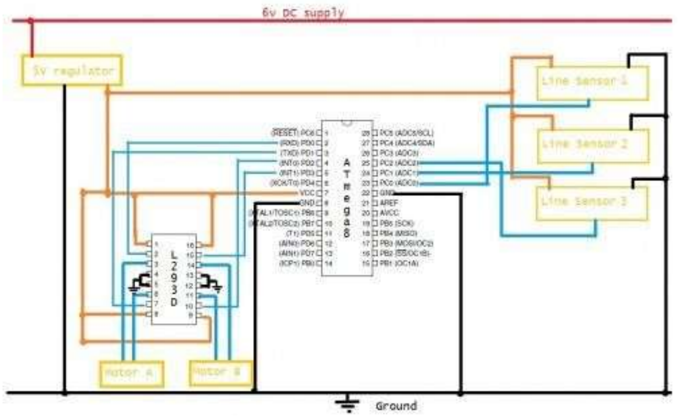
WORKING PRINCIPLE

This Project Grid solving Robot is based on 8 bit Microcontroller AT89S52.This Robot follows the black line which is drawn over the white surface or it follows the white line which is drawn over the black surface. The sensors are used to sense the line. When the light signal falls on the white surface, it gets reflected and if it falls on the black surface, it is not reflected, this principle is used to scan the Lines for the Robot. All the above systems are controlled by the Microcontroller. In our project we are using the popular 8bit microcontroller AT89S52. It is a 28 pin microcontroller. The Microcontroller AT89S52 is used to control the motors. It gets the signals from the sensors and it drives the motors according to the sensor inputs. Two stepper motors are used to drive the robot. The basic principle used for grid following is detecting the intersection and taking a turn accordingly. When a junction is detected we take a single wheel turn, thus aligning the robot on to the right angled path. A small amount of delay is included after each turn to prevent false input to be taken by robot while turning.

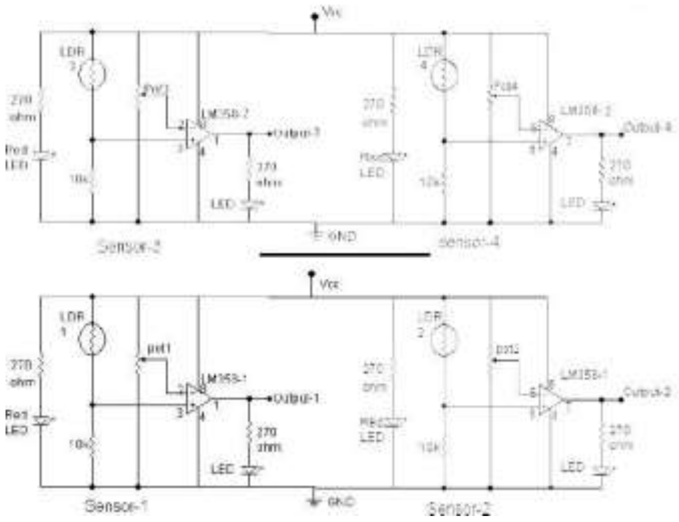
Block Diagram



CIRCUIT DIAGRAM



SENSORS



COMPONENTS

A. Resistors
   - 330 Ω
   - 10K Ω

B. Capacitors
   - 470μF
   - .1μF

C. Sensors

D. OP-AM
   - LM-358

E. LED

F. Microcontroller ATMEGA8

G. Motor driver L2

H. DC Motor(12V,100 rpm)

I. Power Supply (6-12 V)

J. Voltage regulator

K. Variable resistance

L. Connecting wires

M. Chasis

DESCRIPTION

The process of driving to a destination tile always follows the same 5-step-algorithm: (1) calculate the next tile to go to, (2) turn towards the direction of the next tile, (3) start driving and wait for line crossing events to happen, (4) thereby decide which tile the robot arrived in, and (5) finally determine whether the robot has reached the destination tile. The 5-step-algorithm is also visualized in Figure 6.Determining the next tile is done via a breadth-first search: starting from the current tile, the coordinates of all adjacent tiles that have not been visited on the way to the current destination are stored into a vector together with information on its predecessor. If the destination tile is not among the vector entries, again each entry's all adjacent and not yet considered tiles are stored together with their predecessor. The algorithm terminates when the destination tile is in one of the vectors. The path to that destination tile can then be derived using the chain of predecessors stored together with the coordinates. This rather naïve approach has potential to be optimized. An example would be to employ branch-and-bound techniques. Nevertheless, in large grids the applicability of such tree based techniques is limited. In this case greedy algorithms might be an alternative. However, such path finding issues were not the focus of our research Once a path has been derived, the robot turns towards the direction of the next tile (which can be calculated from the current and next coordinates) by turning both wheels in different directions for a predefined time corresponding to 45 degrees (repeatedly if necessary). Because wheels and sensors are attached in a line, the light sensors will not "move" but only rotate.

The robot then starts driving, waiting for events to occur. We distinguish light sensor events and timeout events.

A series of sensor events corresponding to the crossing of the grid) is always concluded by a timeout event. If no sensor events have occurred for a certain time period, it is assumed that the robot has reached the next tile. The length of the interval depends on the size of the tiles, the robot dimensions path deviation and speed. If it is chosen too long (and tiles are small), the robot might already have started to cross the next lines at the opposite side of the new tile. If it is chosen too short, the timeout might occur during the crossing of the grid. For our implementation, we chose a timeout interval of four seconds. Number, kind and order of sensor events determine which tile the robot drove to. Due to path deviations due to wheel slip, different engine powers etc. the robot does not necessarily arrive at the tile it was actually heading for. Basically two ways of crossing the grid can be distinguished: the change to a tile in horizontal (East, West) or vertical (North, South) direction which we will call orthogonal crossing and the change to a tile at north-east, north-west south-east or south-west which we will call diagonal crossing for the rest of this paper. Determining orthogonal crossings is relatively easy: after both sensors have gone dark and light again once and nothing happened afterwards for a while, the robot has reached the next tile. From the order in which the two events occurred a first indication can be drawn on the robot's deviation from the wanted direction (orthogonal to the grid line).

ANALYSIS

The preferred power supply for the circuit is 6V or above but the circuit also works with higher dynamic range if needed, the power supply should be 9V or even 12V. For power supply, you can use an AC-DC wall adaptor, dry batteries or rechargeable batteries.
In this report we presented an approach for mobile robot positioning and navigation which we call grid-based navigation. It uses only minimal environmental infrastructure and two light sensors on the robot. Starting out from a predefined location and orientation in the grid, the mobile robot can autonomously head for destination tiles. On the way it determines its location in the grid using a finite state machine by picking up line-crossing events with its sensors. In addition, we demonstrated how we implemented the underlying algorithm in software and robot hardware. Nevertheless, the required grid limits this approach to scenarios where either a grid can be set up or a natural grid exists (e.g. due to a tiled floor). Nevertheless, it is easy to adjust both sensing and algorithm to different scenarios like a chessboard-like floor. In this paper obstacles are not considered at all. A touch sensor could enable the robot to detect obstacles in tiles which then could be marked as blocked. A major cause of errors is the robot's inaccuracy. Due to its simple construction, the robot shows a hardly predictable behavior in terms of driving and turning speed as well as path deviation when going straight. Thus we found that it does not make much sense to develop more sophisticated algorithmic. approaches. Hence we stuck to simple, qualitative approaches that are robust to such errors. Given a robot that is able to move more exactly and at defined speeds, new possibilities would arise. By measuring the time intervals between light sensor events, the angle in which the robot crosses the grid line could be derive.

AIM

Design and construct an autonomous robot which is capable of traversing the grid by principles of line following and completing the given task. The Task: Reach a sequence of destined destinations (base) within the grid by avoiding the nodes and taking the bridges and then returning to the starting position.

Arena Specifications and Dimensions:

1. Arena will be black shaded with white lines and Black Square in the middle of the path acting as nodal points.

2. No. of grids would be 8x8.

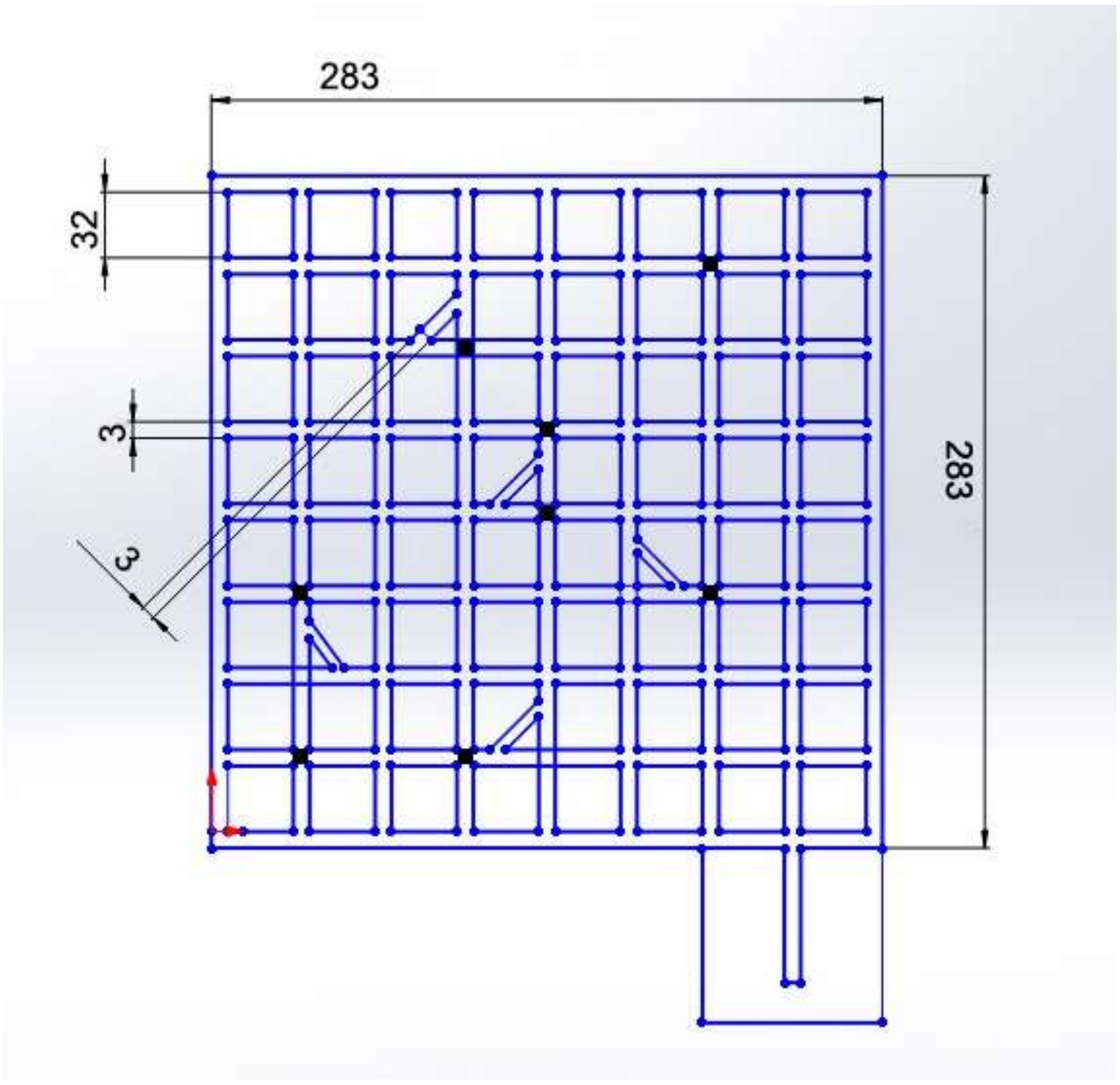
3. Grid dimensions: 35x35 cm

4. Thickness of white lines: 3 cm

5. Thickness of bridge: 3 cm

6. Checkpost are colored in black at different positions.

7. The position of the checkpost may not be as shown in the arena below. The positions of checkpost will be dynamic during the event.



You can find problem statement and other detail at : https://www.ktj.in/events/embetronix

REFERENCES

[1] http://www.scribd.com/doc/35925045/Major-Project-Grid-Solving-Robot#scribd

[2] http://seminarprojects.org/d/grid-follower-c-code-using-2-sensors

[3] http://extremeelectronics.co.in/avr-tutorials/line-following-robot-using-avr-atmega8/

[4] https://www.youtube.com/watch?v=O2Beki9EQoI

[5] https://www.youtube.com/watch?v=7gDoAj7lcqM

[6] http://www.electronicsforu.com/electronicsforu/circuitarchives/view_article.asp?sno=491&article_type=2&id=4679

Contact:-

ANIKET DAGAR
Core Team Head (Finance and Events),
Kshitij 2016 (www.ktj.in)
Technology Students' Gymkhana
IIT Kharagpur
(+91) 8609289019
adagar923@gmail.com
aniket.dagar@ktj.in